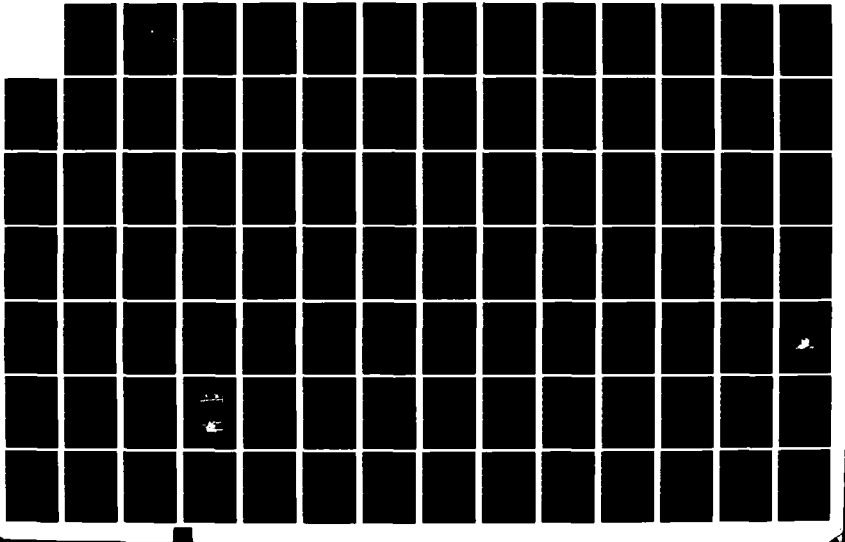
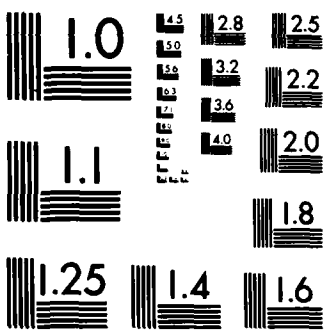
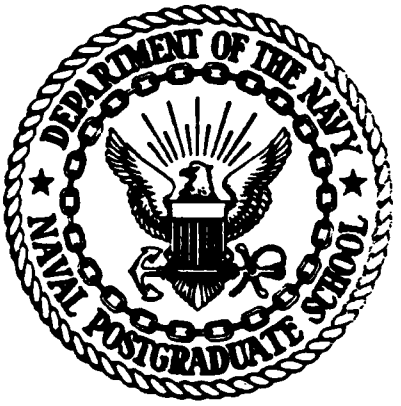
AD-A127 875 UNIVERSAL MICROCOMPUTER INTERFACE FOR DATA ACQUISITION 1/2
(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA W B BROWN
MAR 83

UNCLASSIFIED F/G 9/2 NL





NAVAL POSTGRADUATE SCHOOL
Monterey, California



THESIS

UNIVERSAL MICROCOMPUTER INTERFACE
FOR
DATA ACQUISITION

by

William Burcher Brown, Jr.

March 1983

Thesis Advisor: Donald M. Layton

Approved for public release, distribution unlimited

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): Universal Microcomputer Interface for Data Acquisition	5. TYPE OF REPORT & PERIOD COVERED: Master's Thesis March 1983
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): William Burcher Brown, Jr.	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Postgraduate School Monterey, California 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Postgraduate School Monterey, California 93940	12. REPORT DATE: March 1983
	13. NUMBER OF PAGES: 157
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report)
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

microprocessor, digital control, data acquisition, data logging, TRS-80, Z80, structured programming, BASIC, microcomputer

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A Universal Interface Device (UID) was constructed for the TRS-80 Model I microcomputer that enables it to perform a wide variety of digital and analog data acquisition and control functions within the Department of Aeronautics. The system was tested and validated for low frequency data acquisition and control in a wind tunnel experiment and higher frequency applications were predicted with the development of additional software.

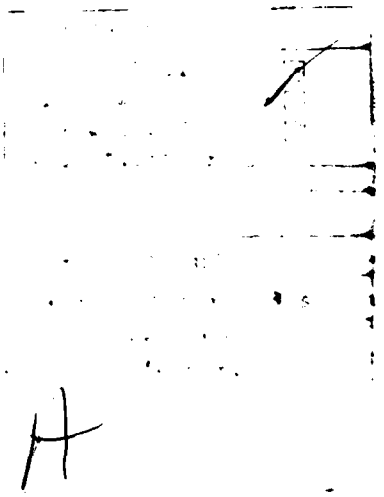
DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 68 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. Abstract Continued

The UID consists of a collection of off the shelf and author designed components assembled within a common enclosure to provide eight channels of analog to digital input, a single channel of digital to analog output, eight bit TTL level digital to digital input and output, and eight power relay outputs all under software control of the host microcomputer.

Applications software, calibration tables, and a validation experiment were prepared and included within the appendices. The UID produced results within one percent of manual observations.



Approved for public release, distribution unlimited

Universal Microcomputer Interface for Data Acquisition

by

William Burcher Brown, Jr.
Lieutenant Commander, United States Navy
B.E.E., Georgia Institute of Technology, 1971

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
March 1983

Author: W Burcher Brown, Jr

Approved by: Donald M Layton
Thesis Advisor

Donald M Layton
Chairman, Department of Aeronautics

J N Dyer
Dean of Science and Engineering

ABSTRACT

A Universal Interface Device (UID) was constructed for the TRS-80 Model I microcomputer that enables it to perform a wide variety of digital and analog data acquisition and control functions within the Department of Aeronautics. The system was tested and validated for low frequency data acquisition and control in a wind tunnel experiment and higher frequency applications were predicted with the development of additional software.

The UID consists of a collection of off the shelf and author designed components assembled within a common enclosure to provide eight channels of analog to digital input, a single channel of digital to analog output, eight bit TTL level digital to digital input and output, and eight power relay outputs all under software control of the host microcomputer.

Applications software, calibration tables, and a validation experiment were prepared and included within the appendices. The UID produced results within one percent of manual observations.

TABLE OF CONTENTS

I. INTRODUCTION

A. BACKGROUND

The age of the microprocessor (U-P) is upon us. The relentless and geometrically accelerating advances in U-P technology are blurring the distinction between large and small computer with quantum leaps in memory capacity and speed of execution. Coincident with these developments has been the increasing application of the U-P first as a purely high cost scientific and research tool and now in low cost consumer electronics and the video game and microcomputer explosions. The microcomputer is the obvious and so far ultimate application of the U-P and is now revolutionizing the workplace and home. It is not outrageous to predict that by the end of the decade, as technological advances continue and economies of scale lower unit prices, the microcomputer should be as common in the workplace as the typewriter and, in the home, as the television.

In regard to scientific data acquisition, the marriage of the microcomputer to the family of concurrently developed analog to digital (A/D) and digital to analog (D/A) integrated circuits has made low cost, easily implemented electronic data acquisition and control an accomplished fact. This paper describes one such application utilizing a TRS-80 Model I microcomputer

interfaced to a collection of commercially available and author designed circuits to provide a portable and universal data collection and automatic control station for the various test instruments located within the Department of Aeronautics at the Naval Postgraduate School (NPS).

B. DISCUSSION

The resulting hardware for the application under discussion has been named the Universal Interface Device (UID) and will be referred to by this name. The host microcomputer is a TRS-80 Model I manufactured by the Tandy Corporation, Fort Worth, Texas and hereafter referred to as TRS-80 and/or Model I. Other models of the TRS-80 are made but are not discussed in this paper. The UID consists of an eight channel analog to digital input port, an eight bit relay output port, an eight bit TTL (transistor-transistor logic) input port, an eight bit TTL output port, and a single digital to analog output port all housed within a common enclosure. Two commercially available, off the shelf devices, the Analog-80 and Interfacer-80, both manufactured by the Alpha Products Company, Woodhaven, New York were used to implement the analog input port, digital input port, and relay output ports. The analog output and digital output ports were constructed from scratch. Subsequent chapters

describe the TRS-80 microcomputer and its operation, the UID and its interface between microcomputer and the outside world, implementation of the combined system for both analog and digital steady state/low frequency I/O (input/output), suggested software routines for various applications, results of preliminary testing of the system, and conclusions and recommendations for further development.

The scope of this discussion assumes some familiarity with microcomputer architecture and digital electronics, but not necessarily a prior knowledge of the TRS-80 system in particular. A comprehensive bibliography is included to direct the user to appropriate topics for review or closer interest. The mastery of the interpreted BASIC high level language resident in the TRS-80 operating system should be a major goal of the user of the UID since this language allows a highly flexible real time interaction with the data acquisition and control process. While it is not possible for this paper to be a comprehensive manual on the BASIC language, suitable attention has been devoted to the subject to provide the user with concepts for developing custom software to drive the UID in a variety of applications. The bibliography references several sources of information for both BASIC and interfacing techniques. Other languages for the TRS-80 are available but not within the scope of this paper; however,

of note for future reference is the fact that the TRS-80 is capable of direct execution of Z80 machine code which would allow high frequency measurements. This application is left for a separate investigation.

The author's purpose is not only to provide an operating manual for the TRS-80/UID system but also stimulate interest and provide a core reference for extended applications of the UID. Thus, it would be an injustice to refer the user to the TRS-80 documentation as the sole source of information for operating the TRS-80 hardware, so there is appropriate attention devoted to describing the operation of the TRS-80, including experiences gained by the author, that are not in the instruction manuals. A certain amount of this is subjective; but, as the project has developed, the author has become highly impressed with the TRS-80 in view of the capabilities implemented, especially since it is generally regarded as a lightweight in professional circles. Of course, no official endorsement of the TRS-80 is intended, but to observe a hobby-grade microcomputer perform as this one has inspired the imagination for the capabilities of those of loftier reputation.

The UID was designed with expansion in mind with both the physical room for additional circuitry and appropriate documentation to allow its future use as a building block for an even more ambitious data acquisition and control station.

II. SYSTEM DESCRIPTION

This section describes the physical and operational characteristics of the TRS-80 Model I and the UID. Where applicable, the various reference manuals and technical documentation listed in the bibliography are cited and will provide additional information beyond the scope of the discussion.

A. TRS-80 MODEL I MICROCOMPUTER

The TRS-80 Model I, first introduced in 1977 by the Tandy Corporation, is a Zilog Z80 microprocessor based microcomputer utilizing an 8 bit word, capable of direct memory access of 64K (kilobytes) of memory, and addressing 256 external ports. It has capacity for up to four 5.25" floppy disk drives, dual cassette drives, a parallel interfaced printer, and a serial RS-232 standard interface in its most standard and complete configuration. The above described peripherals are entirely optional. Of modular design, its main components such as keyboard, CRT monitor, and its various peripherals are separately enclosed and connected to one another by means of ribbon and other multiconductor cables. Resident in the first 16K of memory is the BASIC interpreter in ROM (Read Only Memory) which gives the TRS-80 its built in high level language; the remaining 48K is RAM (Random Access Memory) and is

accessible by the user for either program or data storage. The Model I is frequently referred to in terms of its memory capacity in 16K increments with the first 16K omitted since it is the ROM interpreter and transparent to the user. Thus, the maximum capacity is referred to as 48K even though the Z80 addresses 64K and technically that is what is actually in use. The standard four disk drives are capable of a minimum of 275K of additional storage, and non-OEM (original equipment manufacturer) sources market drives with double density and/or dual heads with capacities extending up to 1M (megabyte) for a four drive installation and hard disk systems capable of up to 10M. The BASIC interpreter is called Radio Shack Level II BASIC and is a variation of Microsoft BASIC, a widely accepted dialect of the BASIC language. Level I BASIC, a distinctly limited subset of Level II BASIC, was also available on some models but is not sufficiently sophisticated for serious application. The standard DOS, the program that interfaces the Model I to its peripheral disk drives, is TRSDOS, now in version 2.3; and other DOS's are available from non-OEM sources.

The TRS-80 Model I is probably the most widely recognized and supported microcomputer in the western world to date. But problems, the most offensive of which were rectified, tarnished its reputation at an early stage and created a somewhat adverse opinion of it as a serious

contender in the business and scientific communities. As an example, early models used an economy keyboard that lacked a professional feel and had severe debounce problems. Likewise, as with many products in the highly competitive computer electronics industry, the hardware was placed on the market before adequate support, both software and service, was available all in the face of an aggressive advertising campaign. The early TRSDOS disk system had a number of bugs that caused great consternation among the first purchasers. Small businesses and professionals expecting instant computing power for the masses were greatly disappointed. Later editions of the Model I, however, corrected these problems with the result that, although identical in outward appearance, they operated almost like a completely different machine. This fact does not seem to be widely known outside the body of dedicated TRS-80 users. But there were other shortcomings, some intentional, that were not corrected including the use of corrosion-prone non-gold plated connectors in the cable assemblies, a non-standard 64 column monochrome CRT dislay, low resolution graphics capability, and the incompatibility with CP/M (CONTROL PROGRAM/MONITOR), the defactor standard microcomputer DOS for business applications [Ref. 1]. As a result, periodic cable disassembly and contact cleaning is necessary, high resolution graphics are not possible, and a great body of

professional software is unusable by the Model I by virtue of its incompatibility with CP/M.

However, limitations aside, the later versions of the Model I are quite capable in their own right; and, because of its mass marketing through the nationwide Radio Shack franchise, the Model I made the microcomputer available on a heretofore undreamed of scale. In contrast, Tandy envisioned the Model I as a strictly experimental product expecting to sell no more than 1000 units during the first year [Ref. 2]. By the end of 1980, however, when production of the Model I ceased due to Federal Communications Commission restrictions on the RFI (radio frequency interference) caused by its unshielded interconnecting cables, over 300,000 units had been sold [Ref. 3] with a multi-million dollar support industry supplying peripheral hardware, software; and two nationally circulated monthly magazines with circulation in excess of 110,000 and devoted exclusively to the TRS-80 were being published [Ref. 4]. Since 1980, the Model I has been replaced by the Model III, a single enclosure unit without the RFI problems of the unshielded Model I but essentially an improved packaging of its design. Although now out of production and obsolete in the face of new products, so much has been and is still being published about the TRS-80 Model I and both the manufacturer and ad hoc user groups encourage and support experimentation and creative applications of it, that it is an ideal platform

on which to base a project such as the UID. A TRS-80 Model I was available in the Department of Aeronautics for this project, as were many salvaged parts used in the construction of the UID. The TRS-80/UID system replaces obsolete and non-functioning data logging equipment in the Department and adds control capabilities not heretofore available with the older equipment. The following subsections describe the various modules which make up the TRS-80 Model I.

1. TRS-80 Model I Architecture

The heart of the TRS-80 Model I is the Zilog Z80 microprocessor which is an MOS technology LSI (large scale integration) device that is manufactured in a standard 40 pin DIP (dual in-line package). Ciarcia, Build Your Own Z80 Computer describes the internal architecture of the Z80 with its pin out which is repeated in Appendix A. The pin-out shows that the Z80 has single power connection, a single clock connection, 8 data lines, 16 address lines, and various control lines as described. The collection of data and address lines are termed the data bus and address bus respectively, and it follows that the 16 address lines can define 2**16 = 65,536 separate and distinct addresses. Likewise, the eight bits of data can be encoded into 2**8 = 256 distinct values within each address. Each data and address line is referred to as a bit and is assigned a corresponding binary weighted value. Eight bits are defined

as a byte and 2**10 = 1025 is defined as a kilobyte or K. Thus 65,536 = 64*(2**10) = 64K, and the Z80 is said to address 64K of on-line memory. The Z80 is also capable of addressing 256 external input or output devices, called ports, via the lower 8 address lines. Data in and out of the Z80 is controlled by pins 19 to 22 which serve to gate the data from or to memory or port addresses [Ref. 5].

The arithmetic operations which the Z80 is able to perform are confined to single and a limited subset of double byte binary integer addition, subtraction, and logical operations with greater range and more complex operations implemented with software. Within the TRS-80, the Z80 chip is connected through buffers to external circuitry. The result is a 40 conductor edge connector termed the TRS-80 bus, or simply bus, for short. Its pin-out is described in Appendix A and a complete schematic of the TRS-80 is found in Radio Shack, TRS-80 Microcomputer Technical Reference Handbook. While the TRS-80 bus is not a pin for pin image of the Z80, it functions essentially the same. It is accessible from the rear of the keyboard unit or the left side of the expansion interface.

The other main feature of the TRS-80 is its collection of memory chips which together comprise the 64K of on-line memory. Since all the Z80 is able to do is add, subtract, compare, and move binary numbers between 0 and

255, a program is required to translate these operations into the decimal arithmetic and alphanumeric character strings that humans can readily understand. In the TRS-80, the first 16K of memory is permanently dedicated to this program which is called the interpreter. The first 12K of the 16 is ROM which, as the name implies, can be read from but not written to. The remaining 4K is RAM which may be written to as well. Within the ROM portion, the interpreter program, in directly executeable Z80 machine code, resides static and unalterable save for chip failure, even when power is not applied. On power-up, the Z80 chip begins to execute the permanent ROM steps starting at address zero. This initializes internal parameters and jump addresses in the 4K of RAM which acts as a scratch pad for the interpreter and ultimately causes the TRS-80 to begin accepting input from the keyboard and from which all other operations are initiated. This is termed the direct or command mode, and from it programs may be entered from keyboard, disk, or tape; execution of an existing program may be initiated by entering the run mode; or an existing program may be altered by entering the edit mode. The syntax accepted by the ROM interpreter is called BASIC and is the familiar high level language common to microcomputers and through which the fairly simple machine/operator communication is accomplished. The remaining 48K of memory in the TRS-80 is RAM; and all programs, variables, and the DOS if a disk system is used,

are stored in it above the first 16K. The TRS-80 will execute BASIC programs using the interpreter and will also directly execute Z80 machine code. Actually, the interpreter translates a BASIC program step by step into Z80 machine code because, as previously explained, that is all the Z80 is capable of executing. Other high level languages, both compiled and interpreted, are available for the TRS-80; but nothing but the resident BASIC interpreter can occupy the lowest 16K of memory. Thus programs in other languages usually require overhead as a run-time module, or in other words, another interpreter, in addition to the program itself.

Minor components of the TRS-80 architecture include the floppy disk controller, printer interface, cassette interface, real time clock, video circuitry, and RS-232 port. Communication between the Z80 and its external components may be done by tying the device to a reserved memory address, called memory mapping; or to a port address, called port mapping. With one exception, all TRS-80 components including the keyboard and CRT display are memory mapped within the previously mentioned 4K of RAM reserved for the interpreter. The exception is the cassette interface which is mapped to port 255. Thus, for example, each character position on the CRT screen is tied to a discrete address between 15360 and 16383 decimal [Ref. 6]; and the keyboard is similarly mapped in a portion of RAM. The

interpreter in a predetermined sequence scans the keyboard memory block, updates the video memory block, triggers the video circuitry, and executes program steps, among other tasks, and then begins again. All of this, including the allocation and management of memory is transparent to the operator. Radio Shack, Level II Basic Reference Manual; Blattner and Mumford, Inside Level II; and Farvour, Microsoft Basic Decoded describe in detail the use of the BASIC interpreter, the mapping of peripherals, and physical layout of the TRS-80.

The TRS-80 Model I system is thus seen to be suitably complex from both a hardware and software standpoint. Through the execution of the BASIC interpreter, the operator is able to control the Z80 microprocessor with the simple English-like commands of BASIC syntax and also use Z80 machine code or combinations of both.

2. Keyboard Enclosure

The TRS-80 keyboard enclosure contains the Z80 microprocessor, interpreter ROM and RAM, supporting circuitry, plus the first 16K of program RAM on a single board located underneath the keyboard assembly. The keyboard is a familiar typewriter-like arrangement with additional keys for the four directional arrows used in carriage control, a clear screen key, and a break key which returns the computer to the command mode when

depressed. Later editions of the Model I have a twelve key numeric keypad to the right of the main keyboard which duplicates the top row of numeric keys. On the side facing away from the operator are located three connectors on the right for connection of the CRT display, power supply, and cassette drive #1; and an on/off button for the keyboard. On the left side of the same face is the 40 conductor edge connector which is the TRS-80 bus. If the expansion interface is installed, this is connected to it by a short ribbon cable; otherwise, it is available for connection of direct connect peripherals. Adjacent to the bus edge card connector and not within direct view of the operator is the reset button which will cause the CPU to reinitialize itself and return to the command mode. The power supply is contained in a separate sealed module external to the keyboard which has connections for the 110 AC power line and a plug with the proper DC voltages to the keyboard. The keyboard assembly connected to the CRT display and power supply is all that is necessary to operate the Model I in its simplest configuration.

3. CRT Monitor

The CRT monitor is a monochrome white on black cathode ray tube with associated circuitry which displays 16 lines in 64 columns (characters). The signal from the keyboard to the monitor is direct video from the character generator within the keyboard unit and is connected by a

short cable and multiconductor plug. Power for the monitor is supplied directly from the AC line by a cord and wall plug. Besides an on/off control, only two other controls, one for contrast and the other for brightness level, are present on the monitor. The monitor resembles a conventional portable television set in both outward appearance and operation.

4. Expansion Interface

The expansion interface is the second half of the Model I system; and, although the keyboard unit and monitor will operate without it, such a system is limited in capacity. The enclosure contains the additional 32K of program RAM, the floppy disk controller, parallel printer interface, and optional RS-232 interface. It is designed to be placed immediately behind the keyboard with the CRT monitor atop it to form a keyboard station of similar dimension and appearance to single enclosure units. Immediately opposite the bus connector on the keyboard is a corresponding connection to the expansion interface which links the two via a short ribbon cable. On the left side face are two additional edge connectors, the rearmost for a parallel standard printer cable; and the foremost, the extension of the computer bus. On the rear face of the left side is still another edge connector for the disk drive cable to the separate and optional four disk drives; and, in the center, sockets for cassette drives 1 and 2.

The socket for drive 1 is in parallel with that on the keyboard unit and provides a convenient place to connect both drives; either can be used. Finally, if the optional RS-232 port is installed, the edge connector protrudes through the center front face.

The expansion interface is powered by an external power supply module identical to that of the keyboard unit. A compartment in the bottom of the expansion interface is provided to house both power supply modules thereby reducing the number of discrete components adrift. Only one control, the on/off button on the front face, is necessary. Once energized, all control eminates from the keyboard.

5. Floppy Disk Drive

Up to four 5.25" floppy disk drive units can be connected to the cable which connects to the rear of the expansion interface. Drives are numbered 0 through 3. 1, 2, and 3 are identical but drive 0 has additional circuitry, termination resistors, required for impedance matching of the transmission line. The disk drive connecting cable has four in-line connectors along its length for attachment to the drive units; any number up to four may be connected. Proper connection of drives is described in Radio Shack, TRSDOS & Disk BASIC Reference Manual. The floppy diskettes are loaded through doors in the front of each drive. Additional software, the DOS, is required to operate a disk

system; and it must be present on the diskette in drive 0 for the system to operate when disks are in use. On power up, the DOS is written into RAM immediately above the 16K interpreter. This reduces the amount of RAM available for program and variable storage, but the increased mass storage of the disk compensates for this loss. The DOS is actually a machine language program which manages the disk space and adds additional commands to the BASIC interpreter. Its operation is transparent to the user and typically requires 10K of overhead if running BASIC; or 5K, if machine language. A number of disk drives of different manufacture are available as are non-OEM DOS's for the TRS-80. Their capacities differ but they operate similarly with the variety of enhancements as their discriminating factor. The DOS causes the disk drive to divide the diskette into concentric tracks typically 35, 40, or 80. Each track is further divided into ten sectors of 256 bytes. The sector is the basic unit of disk space management. Five sectors or half a track is referred to as a gran. Hardware modifications can enable increased data density above the 256 bytes per sector. As an example, an unmodified 40 track system will typically have the capacity to store slightly less than 100K per drive; the exact amount is dependent upon the manner in which the information is written to the disk and the number of files on the disk because a certain amount of the space on each disk is required by the DOS for

housekeeping. Data files may be either random or sequential. Also, the DOS in drive 0 requires approximately 25% of that disk's capacity which reduces the storage available on that drive. There is enough variety in both the disk hardware and software available that it is not practical to go beyond the above general description without unwarranted detail. The user should consult the documentation for the system in use for specific information.

Proper handling of disks and other magnetic media is of prime importance if their operation is to be reliable. Periodic cleaning of disk drive heads, similar to cleaning of audio recording heads; and a clean, grease, smoke, dirt, and electromagnetic field free environment is essential.

6. Tape Drives

The TRS-80 is also capable of using standard audio cassettes for storage of both programs and sequential data files, but the slow speed of tape storage makes it less desireable than a disk system. The bare keyboard unit has provision for one drive; and the expansion interface, a second. Any high quality audio cassette deck may be used and the Model I comes equipped with one as standard. The decks are connected to the computer via supplied cables with a multi-contact plug on the end to the computer and miniature phone jacks to the AUX, EXT, and REMOTE plugs on the tape decks. Operation of the tape decks is straight

forward, and the operator must manually set the transport controls before operation. Since the recorder is an analog device; and the computer, a digital, there is sensitivity in the conversion process particularly in regard to the volume control setting of the tape deck. This is a weak link in the tape system; some experimentation is often required to adjust the volume control for proper tape operation. As with all magnetic media, cleanliness and proper handling of the tape is of prime importance for reliability.

7. Line Printer

A variety of serial and parallel interface printers are available from both the manufacturer and non-OEM sources. Parallel interface printers connect to the previously mentioned slot on the expansion interface and are directly supported in BASIC. Serial printers are connected via the RS-232 port and required additional software drivers. Printers may be dot matrix where the characters are formed by a series of dots imprinted from a moving carriage; or impact, where discrete type, similar to that of a typewriter, creates each character. Many dot matrix characters support the Model I graphics set. Also, modified IBM selectric typewriters are available as are keyboard solenoid arrays which attach to and operate a standard electric typewriter under TRS-80 control. Printer

speeds vary with 80 cps (characters per second) a nominal speed for dot matrix printers; and 25 to 60 cps, for impact printers.

8. Other Peripherals

A wide variety of additional peripherals are available from both the manufacturer and other sources. These include plotters, fast tape systems, analog to digital devices, modems, EPROM programmers, speech synthesizers, voice recognition units, ad infinitum. These products are regularly advertised in the periodicals listed in the bibliography and in Radio Shack catalogs.

9. TRS-80/UID System

At the present writing, the TRS-80 installation available to drive the UID consists of a 16K Model I with 32K expansion interface (total memory = 48K). The keyboard unit is an early model without numeric keypad, and an RS-232 port is installed in the expansion interface. One cassette drive is available for program storage. Disk drives and a printer were borrowed to validate the software for this project and recommendations were made to obtain this additional hardware. All documentation was assembled in a loose leaf binder for ready reference and left in the custody of the Department of Aeronautics laboratory technicians. Appendix B contains various photographs, diagrams, and statistics relative to the TRS-80/UID installation.

B. UNIVERSAL INTERFACE DEVICE

1. General Description

The UID is a collection of circuits housed within a common enclosure that enable a variety of digital and analog I/O from the TRS-80. Those selected were done so after research into the types and characteristics of the applications that would be encountered within the Department of Aeronautics. As part of this, the UID was designed so that its external connectors match those used by the various transducer setups present on the wind tunnels and testing machines within the Department. To maintain compatibility with TRS-80 port notation, all associated port, switch, jack, and connector designations are numbered zero to seven.

The UID enclosure is a standard metal electronics cabinet measuring 22" x 13" x 15". Within it, all circuits are mounted on 44 conductor plug boards mating to sockets mounted within a common card cage. Appendix B contains a table of slot assignments; component orientation is to the right facing the front panel. Connections from the plug board sockets are tied into bundles and run to the front and back panels attaching with quick disconnect push-on connectors. The back panel contains a series of multi-conductor and BNC connectors for convenient connection of signals. The front panel is divided down the middle with the left side containing analog circuitry; and the right,

digital. On the analog side are controls and connections for eight A/D (analog to digital) channels, a single D/A (digital to analog) channel, and a jack and rotary select switch for single point connection of an external meter to monitor any channel in positions zero to seven. The meter jack and selector also provide the output for the D/A channel in position eight. On the digital side are banana jacks for connection to an eight bit TTL level input port, an eight bit TTL level output port, and an eight bit SPDT (single throw double throw) relay port all under control of the TRS-80 host. All digital channels are monitored by panel mounted LED's (light emitting diodes) which show the on or off status of each bit. The center section of the front panel contains ten banana jacks arranged two by five which connect directly to the last two 5 pin connectors on the back panel. With these connectors, non-standard hookups with 5 conductor plugs may be connected directly to the back panel and then patched to the appropriate circuit with jumper cables on the front. Likewise, along the bottom of the front panel are banana jacks which tie to ground, +/- 15 volts, and +5 volts, the standard voltages used within the UID. The UID connects to the TRS-80 by means of a 40 conductor ribbon cable extending from the back panel. Inside the cabinet, the cable continues to a 40 pin DIP jumper which plugs into a mating socket on the port decoder card. Two in-line connectors are also attached to the

internal cable for attachment of the Interfacer-80 and Analog 80 which are designed for direct connection to the TRS-80 bus. The UID power supply is external and connects via four banana jacks on the back panel. Photographs of the front and rear panels are included in Appendix B.

2. Subcomponents

a. Analog Input Port

The analog input port consists of an 8 channel A/D converter, buffer amplifiers, and signal flow and conditioning controls as described in the following:

(1) Analog-80 A/D Converter. The heart of the analog inport port is the Analog-80 A/D converter manufactured by the Alpha Products Company, Woodhaven, New York. This unit is an eight bit resolution, eight channel, port mapped, voltage measuring device with a range from zero to five volts in 20 mV increments and directly connectable to the TRS-80 bus. To adapt it to the UID, it was mounted on a standard 44 conductor plug board with its inputs connected to the plug board connectors and its computer bus ribbon cable and edge connector attached to the internal lead-in ribbon cable from the TRS-80. The Analog-80 port address is jumper selectable to ports 0 to 7; port 0 was selected for the UID application. The Analog-80 only decodes the three least significant bits of the port address so that higher addresses containing a

multiple of the selected port are in conflict and thereby unuseable. The Analog-80 input is unprotected from signals outside of its range of measurement which will cause it catastrophic damage. The Analog-80 receives power from a regulated 9 volt supply on the buffer amplifier board. The manufacturer's operating manual for the Analog-80 is included in the separate documentation binder.

(2) Buffer Amplifier Card. To protect the Analog-80 from under/over voltage, monopolar L8148 OPAMP circuits connected for unity gain were placed between the input signal and the Analog-80 inputs. The OPAMP output signal follows its input signal as long as the input is within the range of the power supply but cannot exceed its supply voltage or drop below ground potential. By providing an adjustable supply for the OPAMP circuits, it is thus possible to prevent the OPAMP output from exceeding the Analog-80 input rating. On the buffer amplifier card, a trim pot is provided to enable the adjustment of this power supply. A second power supply circuit provides +9 volts for operation of the Analog-80 and Interfacer-80; both supplies are powered from the +15 volt external supply.

(3) Input Conditioning Amplifiers. Eight circuit cards with gain selectable OPAMP conditioning circuits are included, one for each A/D channel, to provide for the amplification of small signals to within

the useful range of the Analog-80. Coarse gain is adjustable by a DIP switch located on the amplifier card with fine gain adjustable from precision potentiometers, AMP RANGE 0 to 7, located on the front panel. A second set of potentiometers, AMP BIAS 0 to 7, adjust the signal bias to any level so that an input signal can be conditioned to fit within the 0 to 5 volt dynamic range of the Analog-80 and thus attain maximum resolution.

The input conditioning cards also contain circuitry to drive a diaphram/strain gauge pressure transducer; each has an isolated and regulated +5 volt supply for this purpose. The supply and return signals are connected to the first eight 5 conductor jacks on the back panel. The conditioning amplifiers receive power from the +/- 15 volt external power supply. Conditioning amplifier circuit board pinout and DIP switch gain code are included in Appendix B.

(4) Signal Flow. The analog input port was primarily designed to measure small, millivolt level signals from strain gauge sensing pressure transducers hence the inclusion of the input conditioning amplifiers. However, the capacity to monitor other signals was recognized as desireable, so the signal flow into the A/D converter was designed so the user could inject the signal at a point in the system ahead of the buffer amplifiers. Other devices, such as an analog computer, can thus be utilized

to condition analog input signals as well as the option to not employ signal conditioning at all. Also, recognizing that other A/D devices could be installed in the UID in the future, a DEVICE SELECT rotary switch was included in each channel. The input to the Analog-80 buffer amplifier is tied to position 1 of each of the device select switches; the other unconnected positions are for future expansion. The common pole of the device select switch is connected to the corresponding A/D IN banana jack underneath the device select switch. At the A/D IN jacks an input signal may be injected directly into the analog input port bypassing the internal conditioning amplifiers. Likewise, just underneath the A/D IN jacks are the corresponding AMP OUT jacks which are controlled by the LINE IN switches. With the appropriate LINE IN switch in the down position, the AMP OUT jack is connected to the output of its input conditioning amplifier which is in turn receiving its signal through its five pin connector and a suitable transducer. With the LINE IN switch in the up position, the AMP OUT jack is connected directly to the corresponding BNC connector, on the back panel. To route the AMP OUT signal to the A/D converter, a jumper is inserted between the AMP OUT and A/D IN jacks. This signal flow is illustrated in Appendix B.

b. Analog Output Port

A DAC0801LCN digital to analog converter chip is used in conjunction with a unity gain OPAMP circuit to

provide a D/A output port with a nominal range between zero and five volts in 20 mV increments. The maximum current output of the D/A output port is 10 mA. Greater voltages and power levels must be obtained through external amplification provided by the user. The D/A output port is mapped to port 2 through the bus decoder card described in another section. The D/A output is connected to position 8 of the METER SELECT switch. The D/A output receives its power from the +/- 15 volt and +5 volt external power supply and is colocated with the digital output port. The technical specification sheet for the DAC0801LCN chip is included in the separate documentation binder.

c. Digital Input Port

The digital input port is contained within the Interfacer-80 manufactured by the Alpha Products Company, Woodhaven, New York. It is a TTL level, 8 bit (single byte), port mapped device that returns a decimal integer value between 0 and 255 to the host computer based on the binary weighted value of the signal present in the bit pattern at its input. It is directly connectable to the TRS-80 bus. Also contained within the Interfacer-80 is the relay output port. To adapt the Interfacer-80 to the UID, it was mounted on a standard 44 conductor plug board with its inputs connected to the plug board connectors and its computer bus ribbon cable and edge connector attached to

the in-line connector on the internal lead-in ribbon cable to the TRS-80. The Interfacer-80 port address is jumper selectable to ports 0 through 7; port 1 was selected for the UID. As with the Analog-80, the Interfacer-80 also decodes only the three least significant bits of the port address so that higher addresses containing multiples of the selected port cannot be used.

The digital input port is connected directly to banana jacks TTL IN on the front panel, and panel mounted LED's above the jacks display the status of each bit. The Interfacer-80 is powered by the 9 volt regulated supply on the buffer amplifier card; its operating manual is included in the separate documentation binder.

d. *Digital Output Port*

A modified version of the circuits contained in Titus, TRS-80 Interfacing Book 1 were used to construct a TTL level, 8 bit (single byte), digital output port. The circuitry is colocated with the D/A output port. The output is available at banana jacks TTL OUT on the front panel and LED's monitor the status of each bit. The bit pattern corresponds to the binary weighted value of the decimal output argument. The digital output port is mapped to port 3 through the port decoder card. The port will drive 10 TTL loads (16 mA) and receives power from the +5 volt external power supply.

e. Relay Output Port

The relay output port is contained within the Interfacer-80 previously mentioned and is mapped to port 1. The eight SPDT relays are connected to banana jacks RLY COM (common), RLY NC (normally closed), and RLY NO (normally open) on the front panel. Eight other jacks, LINE IN, are connected to BNC connectors on the back panel. Panel mounted LED's display the status of each relay. The relay contacts may be connected in any configuration through the connecting jacks; maximum ratings are recommended not to exceed 25 volts, 500 mA. The relays are latched according to the binary weighted value of the output argument. The operating manual for the Interfacer-80 is included in the separate documentation binder.

f. Miscellaneous Components

(1) Bus Decoder Card. The analog and digital output ports require additional interfacing to the TRS-80 bus whereas the Analog-80 and Interfacer-80 have their own internal decoding. The bus decoder card is provided for this purpose and is a modified version of the circuits found in Titus, TRS-80 Interfacing Book 1 which not only decode the port I/O but buffer the bus from harmful signals from the interface. The circuit decodes the four least significant bits of the port/address bus for a total of sixteen available ports. The Analog-80 and Interfacer-80

circuits, however, each require two ports, one the designated port address and one made unuseable because of the three significant bits used in decoding; therefore, there are a total of twelve remaining ports available from the bus decoder. Two are used, one for the analog output port; and the other, for the digital output port, with ten remaining for future expansion. The buffered control signals from the TRS-80 bus are likewise available on the bus decoder card. The bus decoder card receives its power from the external +5 volt supply.

(2) LED Indicator Driver Card. A small circuit board using 7404 IC drivers is mounted on the reverse side of the front panel toward the upper center. The LED indicators are driven by these circuits which serve to isolate the LED's from the TTL level I/O signals. The LED indicator driver card is powered by the +5 volt external supply.

(3) Power Supply. An external power supply was used to avoid duplication of effort, and several were available within the Department for this use. The power supply connects to the rear panel via four banana jacks supplying +/- 15 volts, +5 volts, and ground.

III. SYSTEM IMPLEMENTATION

Implementation of the TRS-80/UID system involves five phases: software design, hardware setup, power up, signal calibration, and program execution.

A. SOFTWARE DESIGN

1. General

All port I/O between the TRS-80 and UID is accomplished with the two BASIC statements, INP and OUT, with the following syntax:

```
10 OUT port, arg
20 A = INP (port)
```

Statement 10 is an output statement where "port" is the port address and "arg" the argument sent to the port. Both "port" and "arg" can be a constant, arithmetic expression, or variable. The binary weighted value of "arg" is assigned to the eight bits of "port." Likewise, in statement 20, an input statement, the variable, A, is assigned the decimal value of the signal present at the eight bits of "port." Here "port" can also be a constant, arithmetic expression, or variable. Since the port address and data are both eight bits in length, the values of "port" and "arg" must be between 0 and 255 or an error occurs. Likewise, "port" and "arg" are dimensionless, integer numbers; the relationship between the values passed and the parameters

they represent must be scaled and calibrated by the programmer. In the TRS-80, the INP and OUT statements may be executed in both the command and run modes.

2. Analog Input

Analog input is accomplished by executing the following statements:

```
10 OUT 0, channel
20 A = INP(0)
```

where "channel" is a constant, arithmetic expression, or variable with a value between 0 and 7 corresponding to the desired analog channel to be read. The analog port, as previously explained, is mapped to port 0; a variable or arithmetic expression equal to zero could also have been used in the program statements. In this example, an integer value between 0 and 255 will be returned in the variable, A, corresponding to approximately 20 mV of input voltage per unit of A. A may be converted directly to volts by multiplying by .20. In the general case, the variable, A, will represent some other parameter than voltage; and the scale factor of 20 mV per unit will not be exact due to any number of analog errors in the transmission line. The calibration and scaling of analog input is discussed in a succeeding section.

3. Analog Output

Analog output is accomplished by executing the following statement:

```
10 OUT 2, arg
```

where "arg" may be a constant, arithmetic expression, or variable with a value between 0 and 255. The analog output port, as previously explained, is mapped to port 2; a variable or arithmetic expression equal to 2 could also have been used in the statement. The voltage impressed on the output port will be approximately 20 mV per unit of "arg." As with analog input, the correspondence between the argument value and the output voltage must be calibrated and scaled as described in a succeeding section.

4. Digital Input

Digital input is accomplished by executing the following statement:

```
10 A = INP(1)
```

The digital input port is mapped to port 1, as previously explained; an arithmetic expression or variable equal to 1 could also have been used in the program statement. A decimal integer value will be returned in variable, A, corresponding to the binary weighted value of the bit pattern signal present at the TTL In jacks. Thus, for example, if bit 0 and bit 3 are both above 2.4 volts (corresponding to a TTL level "1" value) and the others are at ground potential (corresponding to a TTL level "0" value) then the value returned in A will be:

```
A = ((2**0)*1) + ((2**3)*1) = 9.
```

The signal present at the digital input port could represent the on/off status of eight binary parameters, a single

binary encoded value, or some other coded combination of parameters; but the value returned in A is a single decimal number. Deciphering the bit status from the decimal value to decode the input is called masking and is required whenever the input parameter represents something other than a single binary encoded value. Masking of digital information is analagous to the calibration and scaling of analog information and is treated in greater detail in a succeeding section.

5. Digital Output

Digital output is accomplished by executing the following statement:

```
10 OUT 3,arg
```

where "arg" may be a constant, arithmetic expression, or variable with a value between 0 and 255. As previously explained, the digital output port is mapped to port 3; an arithmetic expression or variable equal to 3 could also have been used in the program statement. A binary weighted signal corresponding to the decimal value of "arg" will be impressed across the TTL OUT jacks and remain until a similar statement is executed to change it. Thus, as in the preceeding example, if the value of "arg" is 9 then bit 0 and bit 3 will be above 2.4 volts (corresponding to a TTL level "1" value) and the remaining digital output jacks will be at ground potential (corresponding to a TTL level "0" value). The masking of digital output information is

analagous to digital input and is discussed in the aforementioned succeeding section.

6. Relay Output

Relay output is accomplished by executing the following statement:

```
10 OUT 1,arg
```

where "arg" may be a constant, arithmetic expression, or variable with a value between 0 and 255. As previously explained, the relay output port is mapped to port 1; an arithmetic expression or variable with the value 1 could also have been used in the program statement. The action of the relay output port is identical to that of the digital output port except that the output manifests itself at the relay jacks instead of the TTL output. The relays latch according to the binary weighted value of the argument passed and remain so until a similar statement is executed to change them. Thus, as in the two previous examples, if the value of "arg" is 9, then relays 0 and 3 will close; and the rest, open. Masking is also required.

B. HARDWARE SETUP

The UID may be used to monitor any physical parameter that can be converted into an analog or digital signal within the operating range of its various ports. Likewise, the UID can control any device that will respond to a properly conditioned analog or digital signal. BASIC is

suited to steady state applications with time constants on the order of one second or greater. Higher frequency applications would be possible using machine language but is beyond the scope of this investigation. It is the user's responsibility to ensure that the hardware connected to the UID fits within the bandwidth of the system.

Suggested setups for transducer hookup and control applications are listed in Appendix C. Also, a wind tunnel evaluation of a two dimensional NACA 66(215)-216 wing section is also included as Appendix H as part of the validation and testing of the UID. Obviously, the user must be familiar with the signals under investigation, their conditioning, and the logging and control algorithms. Other sections of this paper discuss these topics as do several of the bibliography references.

All hardware connections should be made using good engineering practices and before any power is applied to the TRS-80/UID system. Analog connections must be expecially secure to prevent contact losses. Test runs of the application should be made and monitored with conventional measuring instruments before the TRS-80/UID is energized to ensure no damaging signals will be applied to the UID.

C. POWER UP

The integrated circuitry within the TRS-80 and UID is susceptible to permanent damage from surges in the lines

between components. Inadvertent application of a supply or unusually high signal voltage to the computer bus could have catastrophic results. Likewise, the initial values at the output ports are indeterminate at power up. Devices controlled by the UID could be overdriven by erroneous input signals if energized when power is applied to the TRS-80/UID system. For these reasons, a simple but precise power up procedure is recommended as follows:

1. Make all physical and electrical connections prior to energizing any equipment. Test the set-up to ensure the input signals are safe for the UID.
2. Energize all input devices to the UID.
3. Energize the UID.
4. Energize all TRS-80 peripherals (printer, disk drives, expansion interface, monitor, etc.); a multi-plug AC bus strip is recommended.
5. Energize the TRS-80 keyboard unit.
6. Assign the desired initial values to the digital and analog ports from the keyboard in the command mode.
7. Energize all devices controlled by the UID.
8. Calibrate and scale parameters as required.
9. Load and execute the application program.
10. Power down in reverse order.

D. SIGNAL CONDITIONING

Signal conditioning is the amplification/attenuation and biasing of both input and output signals to levels acceptable to the UID and the devices under its control.

The methods described in this section are illustrated in Appendix C.

1. Analog Input Conditioning

While the analog input port is protected from signals not within its operating range, such signals will be meaningless if accepted as valid data. Likewise, if the dynamic range of the input signal is not of the same order as that of the A/D port, 5 volts, resolution in the A/D process is lost. For these reasons, it is necessary to condition an analog signal.

For analog signals processed through one of the internal conditioning amplifiers, the sensing transducer should be excited to a level slightly less than the expected minimum signal of the application run. This may have to be determined by experimentation. The voltage output from the amplifier is measured and recorded at the AMP OUT jack with a suitable voltmeter. Next the transducer is excited to a level just slightly higher than the maximum expected and this voltage also noted. Then, with the transducer still excited at the higher level, the corresponding AMP RANGE potentiometer is adjusted so the difference between the minimum previously measured and the maximum is within the linear range taken from the channel calibration plots in Appendix G. The transducer is then excited to the former minimum level and the corresponding AMP BIAS potentiometer is adjusted until the signal present at the

AMP OUT jack is slightly above the lower boundary of the linear region. The transducer should be checked again at the higher level and the process repeated until the signals present at the minimum and maximum excitation are within the linear range of the channel. The precise values are not critical as they will be calibrated with software, but it is important that the input signal span the linear range of the A/D channel for maximum resolution.

Signals not processed through the internal amplifiers should be conditioned with external circuitry in a similar manner if not within the range and bias previously mentioned. The user may decide to accept a lesser resolution if the application can tolerate such a condition. An analog computer is a convenient device for external conditioning; Peterson, Basic Analog Computation describes the setup and operation of analog computers that are available within the Department.

2. Analog Output Conditioning

The analog output is capable to supplying 10 mA at five volts and is intended to operate as a dry (low power consumption) control voltage source. An analog computer may be used to condition the analog output signal which may then be amplified by a power amplifier if necessary. Peterson, Basic Analog Computation describes the setup and operation of analog computer available within the Department.

3. Digital Input Conditioning

The TTL logic family is directly cascadable, noise immune, relatively fast (20 MHz typical), and monopolar [Ref. 7] in operation. The high or one state is represented by a signal of 2.4 volts or higher (3.3 typical); and the low or zero state, .8 volts or lower. A +5 volt power supply is standard. Not explicit in the above description is the tendency of TTL connections to "float" or assume the "on" state when left unconnected. This characteristic makes it convenient for the digital input port to monitor mechanical switch contacts because no external excitation voltage is required. With the opposite pole of the contact returned to ground, an open contact represents a high or "1" state; and a closed, a low or "0" state. Such a scheme can be used with the digital input port as can direct connection of other TTL devices. Other logic families may require buffering. Lancaster, TTL Cookbook treats the subject of TTL interfacing in detail.

4. Digital Output Conditioning

TTL devices will source or sink up to 16 mA; that is, the output will supply 16 mA in its on state to an external load returned to ground or will return 16 mA to ground in its off state from an external load supplied from the power source. Thus, from a 5 volt supply, the impedance of any TTL load should not be less than 312 ohms. Many common devices such as LED's (with limiting

resistor), small incandescent lamps, and low power relays fall in this category and can thus be driven directly from a TTL output. Loads requiring higher output can often be driven by transister amplifiers as the TTL level output is also within the range of the typical input signal to a common emitter amplifier. Finally, the TTL level output of the digital output port can directly drive up to ten TTL inputs.

5. Relay Output Conditioning

The output relay contacts of the UID are completely isolated from the internal computer circuitry and can thus be connected in any relay application subject to the limitations listed below. The relays are physically located within the Interfacer-80 enclosure with the contacts rated at 120 volts, 2 amps. The manufacturer, however, does not recommend voltages higher than 25 volts due to the inherent danger of such levels and the potential for catostrophic circuit failure should inadvertent contact be made between a high voltage line and the computer circuitry. The author further recommends limiting the contact amperage to less than 500 mA due to the thin gauge connections between the Interfacer-80 output terminals and the relay output jacks on the front panel. Switching of larger loads can be done through an isolation relay supplied by the user.

E. PROGRAM EXECUTION

Successful program execution is the obviously desired end result of all the above preparations. Proper interfacing and correctly designed software are required, or the old adage of garbage in/garbage out will apply. Therefore, a calibration routine allowing the user to pause in the execution of the program to calibrate and examine the initial response of the TRS-80/UID is always recommended. Accurate calibration will ensure accurate data and control. Likewise, any program should be validated by testing its performance against known results.

Loading and execution of a TRS-80 program is straight forward from the previously cited reference manuals. At the appropriate step in the power up procedure, the program is loaded from tape, disk, or keyboard and execution begun by typing RUN <enter> from the keyboard.

Errors may be encountered during program execution especially from new and untested programs. The user must be aware of the progress of the program and prepared to abort it if problems occur. The first source of error comes from without the system in the form of transducer malfunction or signals exceeding the calibrated range. Parallel monitoring of the input signals with a suitable voltmeter is recommended. The second source of error comes from within the program itself. The most typical are typographical errors in program or data entry, syntax errors

in BASIC usage, mismatch of variables, exceeding the range of a variable, or exceeding the capacity of available memory. The BASIC interpreter contains error handling routines which halt program execution in this case and inform the user of the type of error and the line number in which it occurred. BASIC can also be programmed to internally handle errors or ignore them completely, although this can often be a greater source of program bugs then the error itself. Editing of the program to correct the error is the remedy. The last and an often hidden source of error is incorrect application of the algorithms within the program such that the program executes but does not do what it is supposed to do. Validation of the program against known results is the best source of insurance against this type of problem.

The end result for the TRS-80/UID user will be the successful execution of an applications program. The release from the tedium of manual data collection will allow a more thorough evaluation of the data; and automatic control, greater creativity in the application.

IV. BASIC APPLICATION PROGRAMS

A. GENERAL

Well written software will maximize the interactive capabilities of the TRS-80/UID system and be easier to debug and enhance. A structured, modular approach will result in such programs and has been used in the examples and application programs presented in this section. BASIC is FORTRAN-like in its syntax and, since FORTRAN is a language with which the UID user is likely to be familiar, comparisons between the two have been made. The following sections describe the characteristics of BASIC and present the application software.

B. CHARACTERISTICS OF BASIC

BASIC is an acronym for Beginners All-Purpose Symbolic Instruction Code which was developed at Dartmouth College. As the name implies, it is less formidable than FORTRAN but lacks some its capabilities while adding a few that FORTRAN does not have. In most microcomputers of current design, and the TRS-80 is no exception, BASIC is interpreted rather than compiled. It is still an evolving language with no recognized standard as yet, but its wide acceptance has caused efforts to be made in that direction [Ref. 8]. BASIC's used in microcomputers of different manufacture thus tend to differ slightly in syntax and

minor adaptation is usually required to transport programs from one to the other. The main strength of BASIC is that it is relatively easy to learn and use yet sophisticated enough for serious application. It's weakness lies in its interpretive nature which means that it is relatively slow in execution. What is slow to the computer, however, is often imperceptible to the human senses; and this is often the case. The controversy over BASIC as an acceptable language for sophisticated application rages on and is beyond the scope of this paper. Suffice to say that BASIC is certainly an appropriate medium for the steady state applications of the UID.

BASIC assignment statements, arithmetic expressions, and functions will be recognizeable to the FORTRAN programmer. BASIC supports single and double precision real variables (6 and 14 decimal places) and integer variables but does not support complex variables. These, however, can be synthesized if required. Alphanumeric variables, called string variables, are supported in BASIC where they are not in FORTRAN. The string variable makes possible many applications such as word processing and facilitates interactive English-like communication between the user at the keyboard and the program output on the CRT monitor. Logical and a special set of string operations in BASIC are supported where they are not found in FORTRAN. BASIC supports multi-dimension arrays in all variable

types. BASIC variable labels are two significant characters in length as opposed to the six of FORTRAN and are all global; this can be a minor inconvenience as the selection of label combinations that mnemonically resemble the parameter name are reduced. Thus, the programmer must be careful not to use the same label for a different parameter in another part of the program. Conditional and unconditional transfer statements, IF and GOTO, are also similar to FORTRAN; but the PRINT and INPUT statements replace the WRITE and READ statements of FORTRAN and offer a simpler and wider variety of syntax. Unlike FORTRAN, all BASIC statements must be numbered as the BASIC interpreter uses the line number to locate the program code during execution. BASIC programs are stored as condensed text files in memory above the BASIC interpreter; during execution, the interpreter deciphers each statement as it is encountered and executes a block of its own machine code based on the evaluation of the BASIC statement.

The user is advised to read and gain comprehension of Radio Shack, Level II Reference Manual and Radio Shack, TRSDOS & Disk BASIC Reference Manual for detailed information on the Level II BASIC used in the TRS-80. Both manuals are comprehensive and contain numerous examples illustrating the use of the language. Other references listed in the bibliography may also be useful to the first time BASIC programmer.

C. STRUCTURED BASIC

Structured programming is characterized by easy to follow, step by step program flow with transfer out of the physical statement order not allowed except within defined constructs. Several such constructs are recognized as standard and represent natural steps in decision making and iteration. Examples are: IF-THEN, DO FOR, CASE OF, and others further developed in Appendix D. A group of program statements arranged within these constructs and designed to accomplish a specific purpose is called an algorithm. Program flow enters at the top of the algorithm and exits at the bottom; and, thus, complex program operations may be grouped into algorithms and visualized as discrete modules, a much more manageable method of organizing the program. One construct, the subroutine CALL, causes the program to execute an algorithm located elsewhere within the program and return to the CALLing point when finished; thus transfer of program control in structured programs is not lost, it is just more orderly. Together with internal documentation in the form of REMARK statements, structured programming greatly simplifies the programming task and allows easier debugging and enhancement. All application programs written for the UID have been done so using structured techniques. Graham, Introduction to Computer Science treats the subject of structured programming at great length and is the inspiring source of the methods used in these examples.

The Level II BASIC used in the TRS-80 is not a structured language per se; its various conditional and absolute transfer statements allow the programmer complete freedom in program flow control. However, the structured constructs can be synthesized using combinations of BASIC statements and these translations also appear within the examples of Appendix D. These can be used to build algorithms which obey the logic of the given construct even where there is no BASIC statement that directly applies. Likewise, the indentation of nested constructs in the program listing is a common practice that facilitates readability of the listing but is not an inherent function of Level II BASIC. The program LIST/BAS, Appendix E, has been written to list structured BASIC programs following the indentation conventions and also limits one construct to a page. The program is internally documented and will prompt the user as to its use; it will list to a line printer in structured format any program that has been written following the translations in Appendix D and has been used to supply the listings for the applications software for the UID.

There are, however, two disadvantages to using the above described format with the TRS-80. The first stems from the fact that the added non-executing REMARK statements occupy real memory space which may cause a conflict

in larger programs; any memory byte occupied by a REMARK cannot be used by executing statements or for variable storage. The problem typically becomes acute when large amounts of data must be stored internally in arrays. The solution is to remove the REMARK statements or subdivide the data into smaller modules, often inconvenient. The second disadvantage stems from the manner in which the BASIC interpreter executes transfer statements. Although in theory the structured approach does not use transfer statements, the translation does in synthesizing them; an examination of Appendix D will show this to be evident. When BASIC executes a transfer to a specified line number, the interpreter must search the program text file for it, beginning at the top of the program, because the line numbers are not absolute memory addresses but merely labels of significance only within the program. The label search can occupy perceptible periods of time particularly if the program is long, the label located towards the end of the program, and the transfer executed frequently. The result is a program that executes slowly. The remedy is to place frequently called constructs toward the top of the program.

The user is in no way required to use structured methods with the UID, and at times a judicious combination of structured and non-structured techniques may be optimum. These topics were presented because the application software was

written using them and the user may find the frequently encountered logical patterns of the structured technique of value in BASIC programming efforts. With imagination, the structured approach may be used to produce well documented BASIC programs that execute in reasonable amounts of time.

D. CALIBRATION, SCALING, AND MASKING

1. General

The values passed between the UID and the TRS-80 with the INP and OUT statements are dimensionless integer numbers which require interpretation to be of any value to the program. For example, an A/D input channel with 2.0 volts across it would return approximately the integer value 100 to the program. This corresponds to the 20 mV resolution of the A/D converter and is said to be only approximate because the gain of the buffer amplifier may not be exactly unity and other analog losses may be present. The important thing is that the relationship be linear; determining the scaling factor is the process of calibration. Thus, this value of 100 is at first meaningless in terms of the 2.0 volts that it represents; but if it is multiplied by .02, the estimated scale, the result is 2.0, the measured value. An analogous process exists for digital information where a byte may represent a single binary weighted value, multiple values encoded by some scheme, or the binary status of up to eight discrete

parameters. Deciphering and setting the bit pattern to evaluate or send digital information is called masking. The two following sections develop calibration, scaling, and masking for practical application to the UID.

2. Analog Calibration and Scaling

The A/D process, as previously explained, returns a dimensionless integer value to the computer based on a to be determined scale factor of so many volts per unit of the integer. There may also be present an offset error voltage wherein zero potential across the input channel returns a small but non-zero value to the computer. If the A/D process is assumed to be approximately linear, then these two scaling parameters can be represented by the familiar linear equation:

$$Y = M*X + B$$

where Y is the measured voltage in volts; M, the unknown scale factor in volts per unit of X; X, the dimensionless value returned by the A/D conversion; and B, the unknown offset error in volts. If two samples of X and Y are taken at opposite ends of the dynamic range, the result is two equations in two unknowns which can be solved for M and B. Thereafter, Y can be calculated from any value of X. Likewise, if the measured parameter has units other than volts, then stating the calibration samplings of Y in those units will result in values of M and B that give Y relative to X in that desired unit. The actual gains and

losses in the transmission path between the analog parameter and the A/D converter need never be known as long as the relationship is linear. Operating the A/D converter in the center of its dynamic range will typically give performance that approaches linearity. Likewise, D/A output is essentially the same except the scaling and calibration process is reversed with X as the dependent variable. Specific calibration and scaling algorithms are included in the subroutine library of Appendix F.

3. Digital Masking

If the parameter monitored by a digital port represents a single valued binary weighted bit pattern, then the decimal value sent or returned by the program corresponds directly to the bit pattern; otherwise, masking to decipher the encoded information is required. As an example, two other common encodings which are often encountered are BCD (binary coded decimal) and discrete binary.

BCD requires four bits to represent the ten decimal digits; four bits can represent sixteen values, but BCD only uses the first ten. The least significant bit corresponds to two to the zeroeth power; the next, two to the first; and so on. The combination of the four bits can thus represent 0 to 9, and multiple bit combinations can represent multi-digit decimal numbers. Thus, a single byte

with eight bits can represent two decimal numbers with values from 0 to 9 or a single decimal number with values from 0 to 99.

Discrete binary corresponds to eight separate parameters represented by on/off or yes/no states. There is no weight to the bit pattern since each parameter is separate from the other. Combinations of BCD and discrete binary can also be encoded on a single byte as long as the number of bits required does not exceed the eight available in the byte. Numbers may also be represented in other than BCD format; Hexadecimal (base 16) and Octal (base 8) are two additional examples.

Digital masking in BASIC is directly supported with the AND and NOT operators. If X is the decimal number to be masked and Y the bit position to be determined, then execution of the following statement will return a zero or one in B corresponding to the off/on state of bit Y:

```
10 B = (X AND 2**Y)/2**Y
```

The statement may also be inverted to solve for the value of X that will set bit Y.

The NOT operator returns the complement of the argument and is useful in inverting logic. Whereas the previous example returned a value of one when bit Y was one, the following returns a zero if one and vice versa:

```
B = NOT((X AND 2**Y)/2**Y)
```

Since binary inputs may represent the complement of the desired parameter; the NOT statement is useful for inverting such logic.

Masking algorithms are also included in the subroutine library, Appendix F, for the various examples of this section.

E. SUBROUTINE LIBRARY

A subroutine library has been composed and included as Appendix F. These subroutines are designed to be integrated into any driving program to accomplish their stated operations many of which have been described throughout the text. They include analog and digital I/O through the UID, calibration and scaling of analog information, masking and coding of digital information, delay loops, and a menu driver for interactive programs, and others. The subroutines are written in structured format and use mutually exclusive line numbers; they may be merged into any program as long as the host program line numbers do not conflict with those of the subroutines. A subroutine is called by assigning its input parameters, if required, and executing a GOSUB statement to the appropriate line number. In general, variable labels beginning with the letter V have been used for passing parameters to and from subroutines; and labels beginning with T, for internal calculations. Use of variable labels beginning with

letters other than these two in the host program will avoid inadvertent variable conflict and the hard to find error of having the same label representing two parameters. Each subroutine listing is internally documented with REMARK statements explaining any information necessary for its execution; the REMARK statements may be omitted in an application to conserve memory if desired.

V. TESTING AND VALIDATION

Each circuit of the UID was tested for proper operation and the results included within Appendix G. To demonstrate the TRS-80/UID in practical operation, the system was connected to the 32" x 45" academic wind tunnel in Halligan Hall and used to evaluate a NACA 66(215)-216 two dimensional wing section. The results are included as Appendix H.

A. TESTING

Testing of the UID circuits involved separate techniques for the analog and digital circuits. For the digital, the operation of each bit could be observed from the LED indicators mounted on the front panel; and each of the three circuits responded by setting the correct bit pattern for the entire range of 8 bit I/O possible. A truth table for the digital circuits is included in Appendix G. For the analog circuits, the analog output was first excited through the entire argument range of 0 to 255 and the resulting output voltage measured, tabulated, and plotted. The calibrated analog output was then used to drive each analog input channel and the corresponding results tabulated and plotted for each channel. These tables and calibration curves are included in Appendix G.

B. VALIDATION

The TRS-80/UID was used to investigate the NACA 66(215)-216 wing section in the following manner. The test wing section with 36 predrilled pressure taps distributed over its surface was mounted in the wind tunnel and the port taps connected to the tunnel mounted scanivalve/pressure transducer and the tunnel manometer array. The relay output port was used to drive the scanivalve controller; and the digital input port, to monitor the BCD encoded scanivalve address. An EAI TR-20 analog computer was used to condition the output from the pressure transducer whose output was applied to channel 0 of the analog input port. A driving program which included the chord/camber/pressure tap coordinates of the test section in internal statements was composed to first calibrate the UID to the pressure transducer and then record the pressure field around the test section by stepping the scanivalve through its ports. Coincident with the TRS-80/UID run, a photograph of the manometer array was taken and the same pressure field tabulated manually. The entire report is included in Appendix H. Agreement of the comparisons was within one percent.

VI. RESULTS AND CONCLUSIONS

The TRS-80/UID has been described and demonstrated in practical application in the previous sections for steady state data logging and low frequency control applications. Included are calibration curves for the analog circuits and truth tables for the digital as well as a demonstration of a practical application with the wind tunnel evaluation. The analog calibration curves show linear behavior and the manual and UID tunnel observations agreed within one percent. The TRS-80/UID is considered validated for the above use.

The system is capable of expansion, and ten additional port addresses are easily accessible from the bus decoder card. Of particular utility would be a multi-byte resolution A/D channel for greater accuracy in the analog input port which the author had hoped to implement but regrets that time constraints forced him to abandon. Likewise, high frequency data logging is also a possibility with the use of machine language programs, and that capability remains to be investigated. The applications are only limited by the imagination.

The TRS-80/UID project has been a true challenge. In building it, the author hopes to have provided a powerful and useful test instrument for the Department of Aeronautics

and also has realized a large amount of personal satisfaction in working with the equipment and expounding ideas of keen interest. As demonstrated, the microprocessor is revolutionizing the world in which we live.

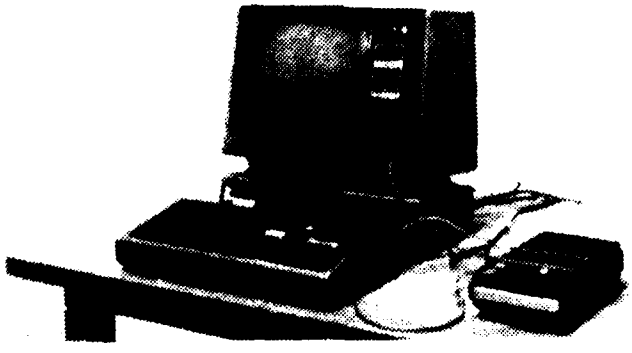
APPENDIX A

SPECIFICATIONS

TRS-80 MODEL I MICROCOMPUTER

Manufactured by

Tandy Corporation
One Tandy Center
Fort Worth, Texas 76102



TRS-80 Model I System

Z80 Microprocessor Pinout

Pin No.	Mnemonic	Nomenclature	Function
1	A11	address bus	bit 11
2	A12	address bus	bit 12
3	A13	address bus	bit 13
4	A14	address bus	bit 14
5	A15	address bus	bit 15
6	CLOCK	clock	system timing
7	D4	data bus	bit 4
8	D3	data bus	bit 3
9	D5	data bus	bit 5
10	D6	data bus	bit 6
11	+5V	+5 volts	power supply
12	D2	data bus	bit 2
13	D7	data bus	bit 7
14	D0	data bus	bit 0
15	D1	data bus	bit 1
16	INT*	interrupt	CPU control
17	NMI*	non-maskable interupt	CPU control
18	HALT*	halt	CPU control
19	MREQ*	memory request	system control
20	IORQ*	input/output request	system control
21	RD*	memory read	system control
22	WR*	memory write	system control
23	BUSAK*	bus acknowledge	bus control
24	WAIT*	wait	CPU control
25	BUSRQ*	bus request	bus control
26	RESET*	reset	CPU control
27	M1*	memory cycle one	system control
28	RFSH*	refresh	system control
29	GND	ground	signal reference
30	A0	address bus	bit 0
31	A1	address bus	bit 1
32	A2	address bus	bit 2
33	A3	address bus	bit 3
34	A4	address bus	bit 4
35	A5	address bus	bit 5
36	A6	address bus	bit 6
37	A7	address bus	bit 7
38	A8	address bus	bit 8
39	A9	address bus	bit 9
40	A10	address bus	bit 10

Reference: Ciarcia, Build Your Own Z80 Computer

TRS-80 Model I Microcomputer

Bus Pinout

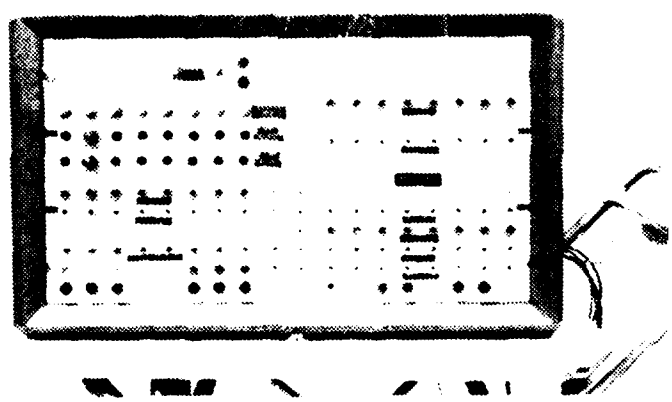
Pin No.	Mnemonic	Nomenclature	Function
1	RAS*	row address strobe	memory refresh
2	SYSRES*	system reset	system control
3	CAS*	column address strobe	memory refress
4	A10	address bus	bit 10
5	A12	address bus	bit 12
6	A13	address bus	bit 13
7	A15	address bus	bit 15
8	GND	ground	signal reference
9	A11	address bus	bit 11
10	A14	address bus	bit 14
11	A18	address bus	bit 18
12	OUT*	peripheral write	system control
13	WR*	memory write	memory control
14	INTAK*	interupt acknowledge	system control
15	RD*	memory read	memory control
16	MUX	multiplexor control	memory control
17	A9	address bus	bit 9
18	D4	data bus	bit 4
19	IN*	peripheral read	system control
20	D7	data bus	bit 7
21	INT*	interupt input	system control
22	D1	data bus	bit 1
23	TEST*	test	system testing
24	D6	data bus	bit 6
25	A0	address bus	bit 0
26	D3	data bus	bit 3
27	A1	address bus	bit 1
28	D5	data bus	bit 5
29	GND	ground	signal reference
30	D0	data bus	bit 0
31	A4	address bus	bit 4
32	D2	data bus	bit 2
33	WAIT*	processor wait	memory control
34	A3	address bus	bit 3
35	A5	address bus	bit 5
36	A7	address bus	bit 7
37	GND	ground	signal reference
38	A6	address bus	bit 6
39	+5V	+5 volts	power supply
40	A2	address bus	bit 2

Reference: Radio Shack, TRS-80 Microcomputer Technical Reference Manual

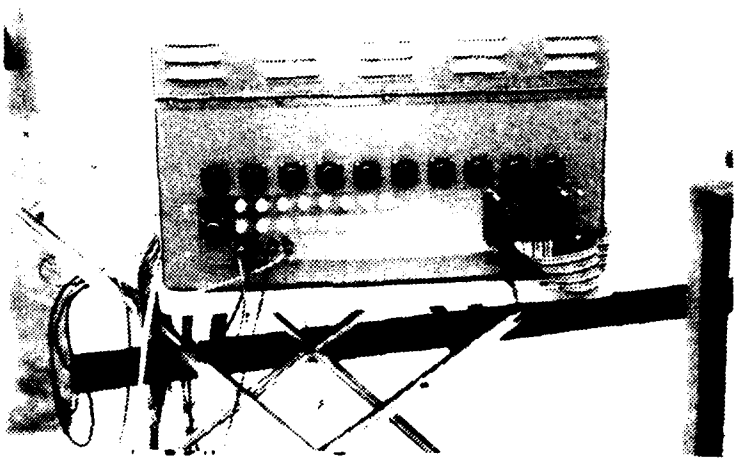
APPENDIX B

SPECIFICATIONS

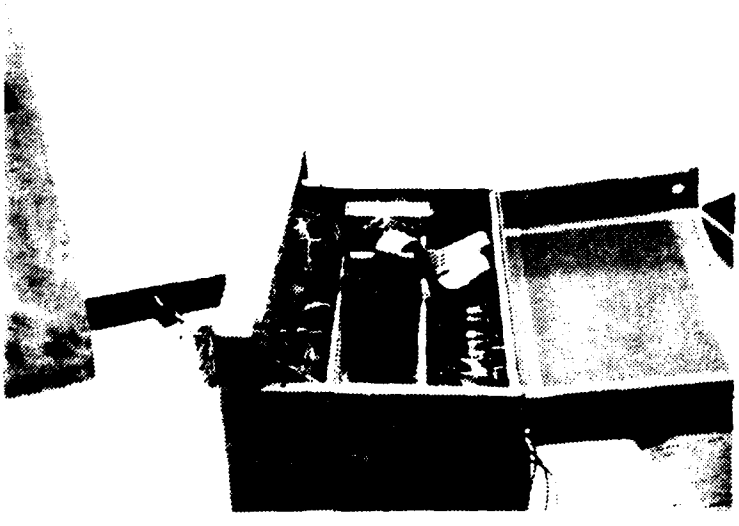
UNIVERSAL INTERFACE DEVICE (UID)



UID Front Panel



UID Rear Panel



UID Interior

Universal Interface Device

Circuit Card Slot Index

Slot	Assignment
1	analog conditioning amplifier channel 0
2	analog conditioning amplifier channel 1
3	analog conditioning amplifier channel 2
4	analog conditioning amplifier channel 3
5	analog conditioning amplifier channel 4
6	analog conditioning amplifier channel 5
7•	analog conditioning amplifier channel 6
8	analog conditioning amplifier channel 7
9	buffer amplifier/aux power
10	Analog-80
11	Interfacer-80
12	bus decoder
13	A/D & D/D output
14	future expansion
15	future expansion
16	future expansion

Universal Interface Device

TRS-80 Port Map of Components

Port	Assignment
0	A/D input port (Analog-80)
1	D/D input and relay output port (Interfacer-80)
2	analog output port
3	digital output port (TTL)
4	available
5	available
6	available
7	available
8	not used (bus conflict with Analog-80)
9	not used (bus conflict with Interfacer-80)
10	available
11	available
12	available
13	available
14	available
15	available

Universal Interface Device

Component Pinout

Conditioning Amplifier Card

Card Slots 1 to 8

1-A.
2-B. ground
3-C.
4-D. EO (output)
5-E.
6-F. AMP BIAS (wiper + high)
7-H. AMP BIAS (low)
8-J. E1 (input)
9-K.
10-L.
11-M. E2 (input)
12-N.
13-P. AMP RANGE (wiper)
14-R. AMP RANGE (gnd)
15-S. +5V (transducer excitation)
16-T. AMP RANGE (high)
17-U.
18-V.
19-W.
20-X. +15V (supply)
21-Y. ground
22-Z. -15V (supply)

Universal Interface Device

Component Pinout

Analog-80

Card Slot 10

1.		A.	+9V (supply)
2.		B.	
3.	ADINP 0	C.	
4.		D.	
5.	ADINP 1	E.	
6.		F.	
7.	ADINP 2	H.	
8.		J.	
9.	ADINP 3	K.	
10.		L.	
11.	ADINP 4	M.	
12.	ADINP 5	N.	
13.		P.	
14.	ADINP 6	R.	
15.		S.	
16.	ADINP 7	T.	
17.		U.	
18.	+5.12 reference	V.	
19.		W.	
20.	ext strobe	X.	
21.		Y.	
22.		Z.	ground

Universal Interface Device

Component Pinout

Interfacer-80

Card Slot 11

1.	+9V (supply)	A.	
2.	RLY COM 1	B.	RLY COM 5
3.	RLY NC 0	C.	RLY COM 4
4.	RLY NO 0	D.	RLY NO 4
5.	RLY COM 0	E.	RLY NC 4
6.	RLY COM 7	F.	LED 7
7.	RLY NO 7	H.	LED 6
8.	RLY NC 7	J.	LED 5
9.	RLY COM 6	K.	LED 4
10.	RLY NO 6	L.	LED 3
11.	RLY NC 6	M.	LED 2
12.	RLY COM 5	N.	LED 2
13.	RLY NO 5	P.	LED 0
14.	TTL INP 7	R.	RLY NC 3
15.	TTL INP 6	S.	RLY NO 3
16.	TTL INP 5	T.	RLY COM 3
17.	TTL INP 4	U.	RLY NC 2
18.	TTL INP 3	V.	RLY NO 2
19.	TTL INP 2	W.	RLY COM 2
20.	TTL INP 1	X.	RLY NC 1
21.	TTL INP 0	Y.	RLY NO 1
22.	ground	Z.	

Universal Interface Device

Component Pinout

Bus Decoder

Card Slot 12

1.		A.	+5V (supply)
2.	PORTSEL* 0	B.	
3.	PORTSEL* 1	C.	D7
4.	PORTSEL* 2	D.	D6
5.	PORTSEL* 3	E.	D5
6.	PORTSEL* 4	F.	D4
7.	PORTSEL* 5	H.	D3
8.	PORTSEL* 6	J.	D2
9.	PORTSEL* 7	K.	D1
10.	PORTSEL* 8	L.	D0
11.	PORTSEL* 9	M.	
12.	PORTSEL* 10	N.	INT*
13.	PORTSEL* 11	P.	RD*
14.	PORTSEL* 12	R.	INTAK*
15.	PORTSEL* 13	S.	WR*
16.	PORTSEL* 14	T.	OUT*
17.	PORTSEL* 15	U.	IN*
18.	INPREQ* 0	V.	RESET*
19.	INPREQ* 1	W.	
20.	INPREQ* 2	X.	
21.	INPREQ* 3	Y.	
22.		Z.	ground

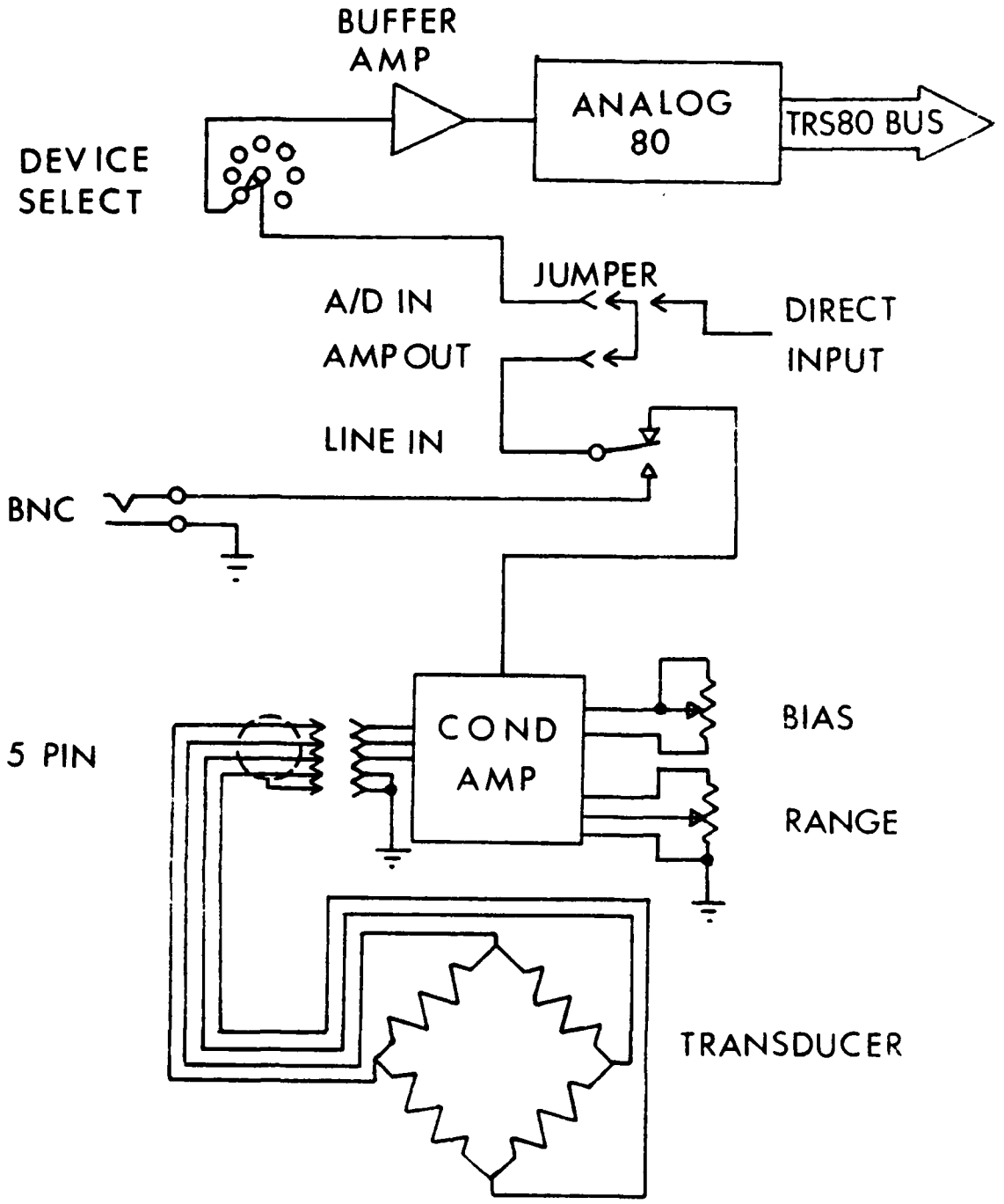
Universal Interface Device

Component Pinout

D/A & D/D Output Ports

Card Slot 13

1.		A.	+5V (supply)
2.		B.	+15V (supply)
3.		C.	-15V (supply)
4.	TTL 0	D.	D0
5.	TTL 1	E.	D1
6.	TTL 2	F.	D2
7.	TTL 3	H.	D3
8.	TTL 4	J.	D4
9.	TTL 5	K.	D5
10.	TTL 6	L.	D6
11.	TTL 7	M.	D7
12.		N.	PORTSEL* 2 (D/A output)
13.		P.	
14.		R.	
15.		S.	
16.		T.	PORTSEL* 3 (D/D output)
17.		U.	
18.		V.	D/A OUT (0 to 5V)
19.		W.	
20.		X.	
21.		Y.	
22.		Z.	ground



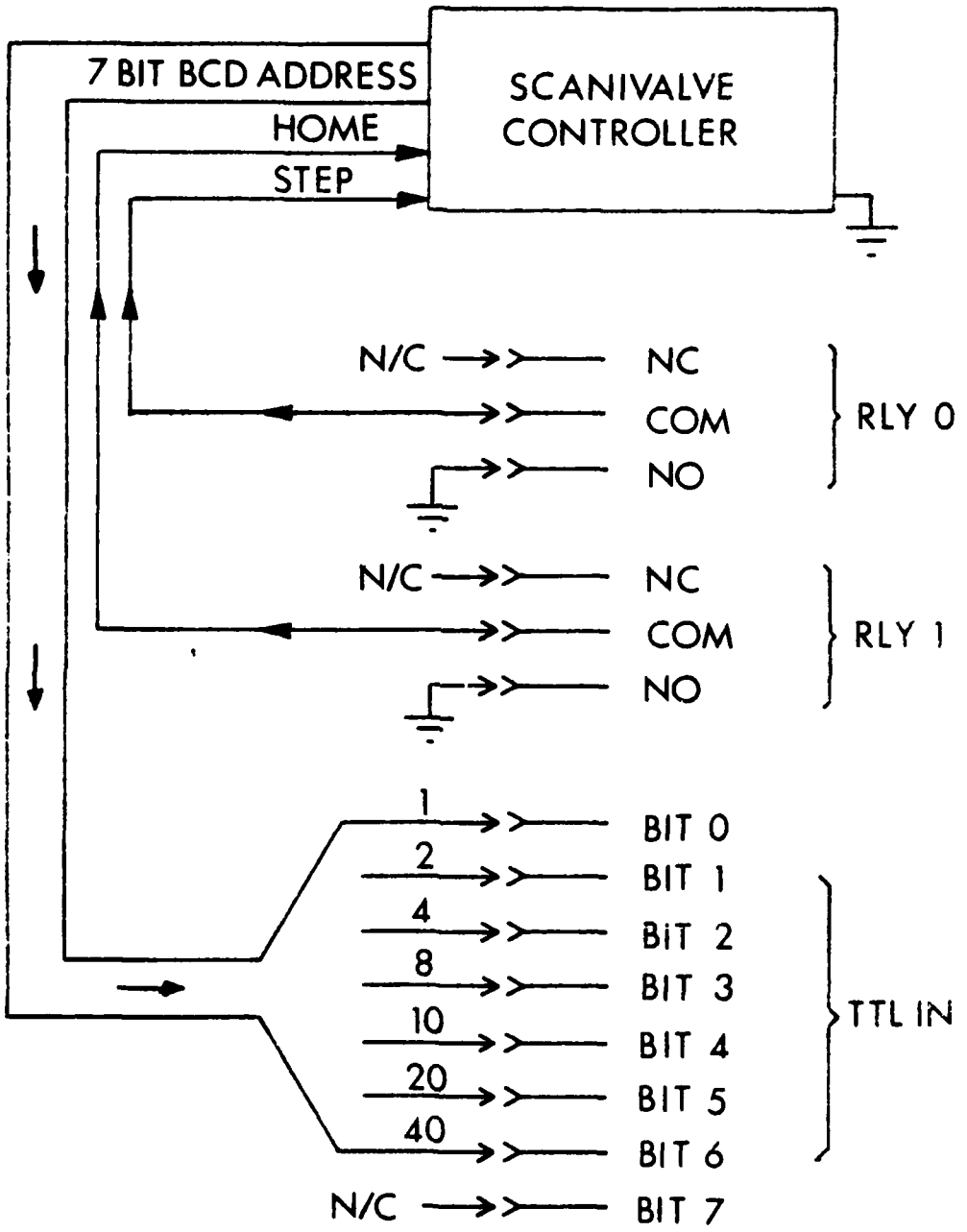
A/D CHANNEL SIGNAL FLOW
TYPICAL

APPENDIX C

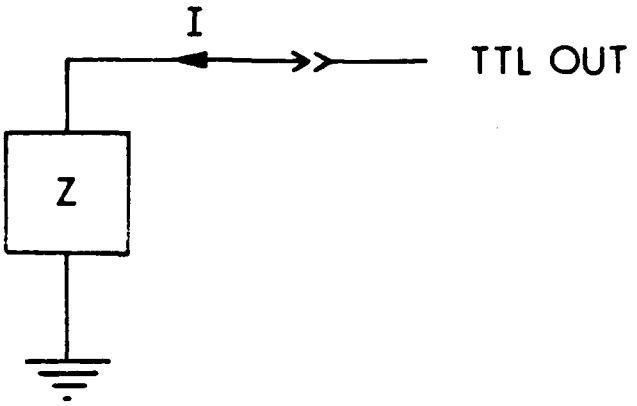
HARDWARE APPLICATION DIAGRAMS

The following pages contain illustrations for recommended applications of the TRS-80/UID system. They are:

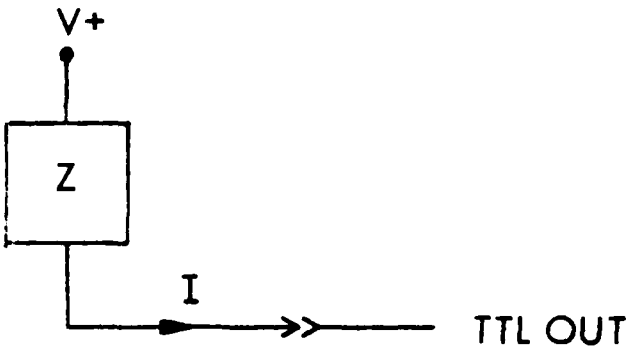
1. scanivalve control
2. TTL output
3. TTL input
4. A/D channel external signal conditioning



SCANIVALVE CONTROL
SETUP



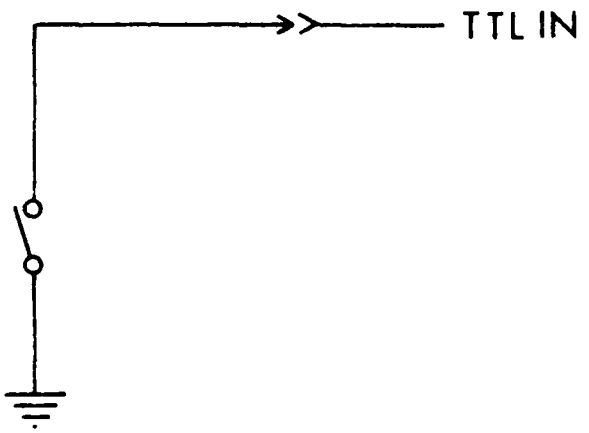
HIGH STATE = LOAD ENERGIZED



LOW STATE = LOAD ENERGIZED

I < 16 mA
Z > 312 ohms

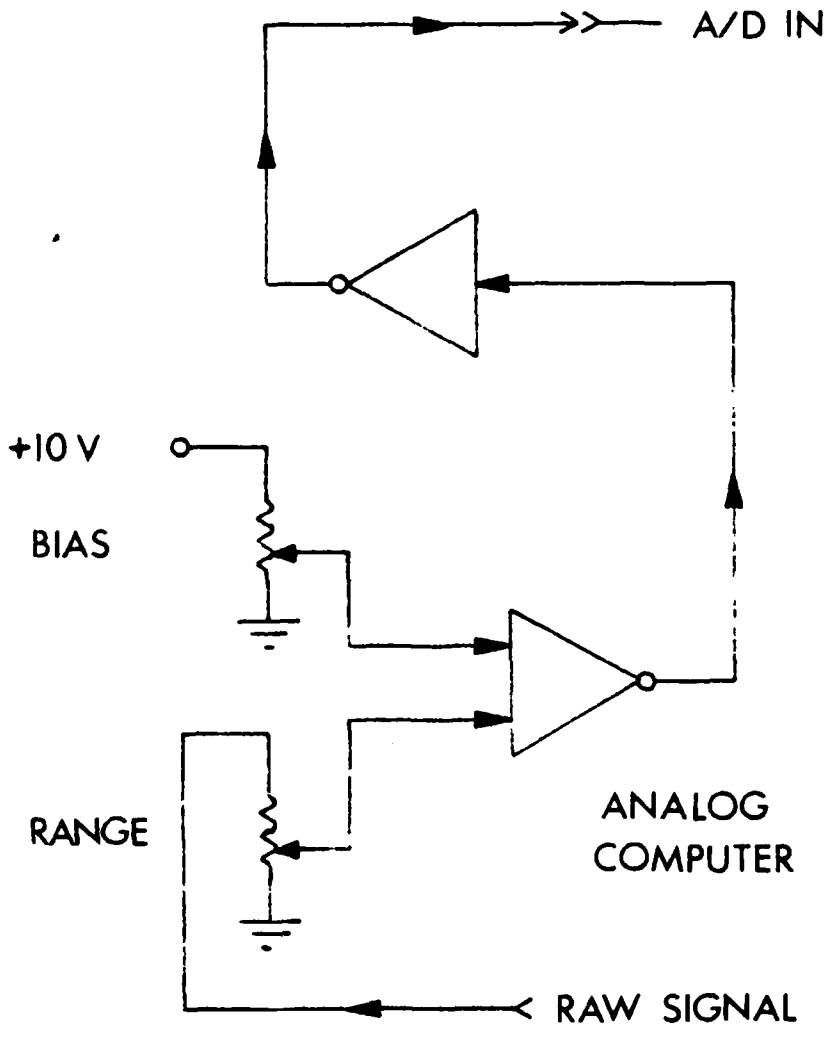
TTL OUTPUT
SETUP



CONTACT OPEN = HIGH STATE
CONTACT CLOSED = LOW STATE

NO EXCITATION POWER REQUIRED

TTL INPUT
SETUP



A/D CHANNEL
EXTERNAL SIGNAL CONDITIONING
SETUP

APPENDIX D

STRUCTURED BASIC

The following pages describe the logical constructs of structured programming and list skeletal methods for implementing them in BASIC. They are:

1. IF-THEN-ELSE (conditional execution)
2. DO FOR (unconditional iteration)
3. DO UNTIL (conditional iteration)
4. DO WHILE (conditional iteration)
5. CASE OF (conditional execution)
6. LOOP UNTIL (conditional iteration)
7. CALL (unconditional transfer)

IF-THEN-ELSE

Executes the succeeding statement(s) if the given condition is true. Optional multiple conditions and default ELSE. The first set of statements whose condition is true are executed (or the ELSE statements) and none others.

Structured Abstraction

```
IF (condition 1) THEN
    (program statements)
ELSE IF (condition 2) THEN
    (program statements)
ELSE IF ........
              .
              .
              .
ELSE IF (condition n) THEN
    (program statements)
ELSE
    (program statements)
END IF
```

BASIC Translation

```
100  'If (condition 1) THEN
102       IF (NOT(condition 1))THEN 110
               (program statements)
109       GOTO 199
110  'ELSE IF (condition 2) THEN
112       IF(NOT(condition 2))THEN 120
               (program statements)
119       GOTO 199
120  'ELSE IF ........
                   .
                   .
                   .
1NO  'ELSE IF (condition n) THEN
1N2       IF(NOT(condition n))THEN 190
               (program statements)
1N9       GOTO 199
190  'ELSE
               (program statements)
199  'END IF
```

DO FOR (index) = (p1) TO (p2) STEP (p3)

Executes the statements within the construct a given number of times determined by setting the index to parameter 1 and incrementing (decrementing) it by parameter 3 for each iteration until ABS(index) >= ABS(parameter 2). Index may be used on right, but not left, side of an assignment statement within the construct. Index may be a real or integer simple variable (no arrays); parameters may be real/integer simple or array variable, arithmetic expression, or constant. STEP (parameter 3) is optional; '1' is default.

Structured Abstraction

```
DO FOR (index) = (param 1) TO (param 2) STEP (param 3)
    (program statements)
END DO
```

BASIC Translation

```
100  'DO FOR (index) = (param 1) TO (param 2) STEP (param 3)
102       FOR (index) = (param 1) TO (param 2) STEP (param 3)
              (program statements)
198       NEXT (index)
199  'END DO
```

DO UNTIL (condition)

Executes the statements within the construct until the condition is true. Executes at least once even if condition is initially false. Assignment statement within loop must cause condition to converge to true or infinite loop will result.

Structured Abstraction

```
DO UNTIL (condition)
        (program statements)
    (condition) <= TRUE
        (program statements)
END DO
```

BASIC Translation

```
100  'DO UNTIL (condition)
             (program statements)
1XX        (condition) <= TRUE
             (program statements)
198        IF(NOT(condition))THEN 100
199  'END DO
```

DO WHILE (condition)

Executes the statements within the construct while the condition is true; skips the construct altogether if condition is initially false. Assignment statement within loop must cause condition to converge to true or infinite loop results.

Structured Abstraction

```
DO WHILE (condition)
        (program statements)
    (condition) <= TRUE
        (program statements)
END DO
```

BASIC Translation

```
100  'DO WHILE (condition)
102     (IF(NOT(condition))THEN 199
            (program statements)
1XX      (condition) <= TRUE
            (program statements)
198     GOTO 100
199  'END DO
```

CASE OF (index) = 1 TO (constant)

Executes the index'th set of statements within the construct or optional ELSE default. Index must be integer variable and constant be integer greater than '1'.

Structured Abstraction

```
CASE OF (index) = 1 TO (constant)
CASE 1:
    (program statements)
CASE 2:
    (program statements)
CASE 3: . . . . . .
              .
              .
              .
CASE (constant):
    (program statements)
ELSE:
    (program statements)
END CASE
```

BASIC Translation

```
100  'CASE OF (index) = 1 TO (constant)
102       IF (index) < 1 OR (index) > (constant) THEN 190
104       ON (index) GOTO 110,120, . . . ,1C0
110  'CASE 1:
               (program statements)
119       GOTO 199
120  'CASE 2:
               (program statements)
129       GOTO 199
130  'CASE 3: . . . . . .
                        .
                        .
                        .
1C0  'CASE (constant):
                (program statements)
1C9       GOTO 199
190  'ELSE:
                (program statements)
199  'End case
```

LOOP UNTIL (condition)

Executes the statements within the construct until condition is true. Exits the loop immediately upon a true evaluation of condition (unlike the DO constructs which exit at the bottom). Assignment statement within loop must converge to true condition or infinite loop will result.

Structured Abstraction

```
LOOP UNTIL (condition)
        (program statements)
    (condition) <= TRUE
        (program statements)
    IF (condition) = TRUE THEN QUIT
        (program statements)
END LOOP
```

BASIC Translation

```
100  'LOOP UNTIL
              (program statements)
1XX         (condition) <= TRUE
              (program statements)
1YY        IF (condition) THEN 199:'QUIT
              (program statements)
198        GOTO 100
199  'END LOOP
```

CALL (subroutine)

Transfers program control to subroutine and resumes execution at CALLing point when subroutine is ENDed.

Structured Abstraction

```
ALGORITHM MAIN PROGRAM
        (program statements)
    CALL (subroutine) (parameters)
        (program statements)
END MAIN PROGRAM
                .
                .
                .
SUBROUTINE (subroutine) (parameters)
        (program statements)
END (subroutine)
```

BASIC Translation

```
100  'ALG MAIN PROGRAM
            (program statements)
1XX       (parameters) <= (calling parameters)
1YY      GOSUB 200:'(subroutine)
1ZZ       (returning parameters) <= (parameters)
            (program statements)
198      END
199  'END MAIN PROGRAM
                    .
                    .
                    .
200  'SBR (subroutine)
202      (local parameters) <= (calling parameters)
            (program statements)
297      (returning parameters) <= (local parameters)
298      RETURN
299  'END (subroutine)
```

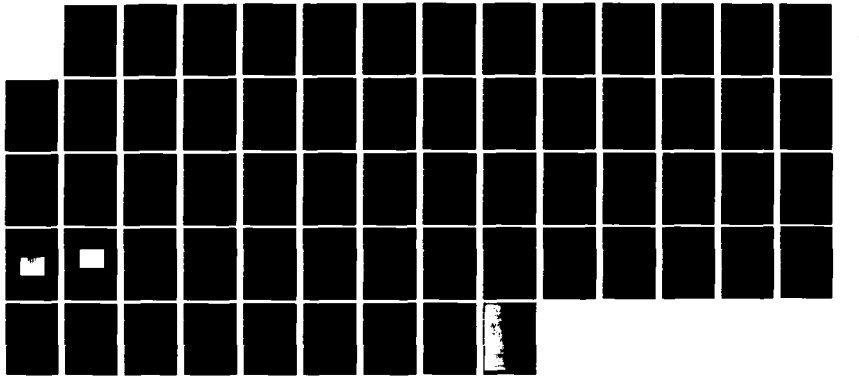
```
  1 '*****************************************************************
  2 '***                                                           ***
  3 '***                    STRUCTURED LISTING                     ***
  4 '***                                                           ***
  5 '***                 BY LCDR W. B. BROWN, JR                   ***
  6 '***                    28JUL82/REV 1.0                        ***
  7 '***                                                           ***
  8 '*****************************************************************
  9 '
 10 '
 20 CLEAR 5000

100 'ALG PROGRAM DRIVER
102       CLS
104       PRINT"THIS PROGRAM WILL LIST THE PROGRAM STORED"
106       PRINT"AS 'TEMP/BAS' AS A TEXT FILE IN STRUCTURED"
108       PRINT"FORMAT."
110       PRINT
112       INPUT"ENTER TITLE";TITLE$
114       PRINT
120       INPUT"ENTER LOWER AND UPPER LINE NUMBERS INCLUSIVE (XX,YY)";X
          Y
122       PRINT:PRINT
123       INPUT"ENTER BEGINNING PAGE NUMBER";PN:IF PN<1 THEN PN=0

124       GOSUB 1000:'MAIN PROGRAM
126       LPRINT CHR$(140);STRING$(35,13)
128       PRINT:PRINT:PRINT:PRINT:PRINT
130       PRINT"JOB DONE"
132       PRINT
134       END
197 'END ALG
```

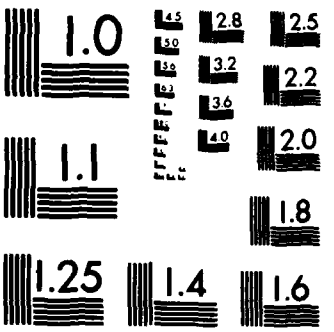
```
1000 'SBR MAIN PROGRAM
1010       GOSUB 2000:'INITIALIZE
1020       OPEN "I",1,"TEMP/BAS"
1030       'DO UNTIL EOF(1) OR NL>=YL
1040            LINE INPUT#1,T$
1050            T=INSTR(T$," ")
1060            LN$=LEFT$(T$,T-1)
1070            NL=VAL(LN$)
1080            LN$=RIGHT$(T$,LEN(T$)-T)
1082            'IF NL>=XL THEN
1084                 IF(NOT(NL>=XL))THEN 1242
1090                 'DO FOR J=1 TO JN
1100                      FOR J=1 TO JN
1110                      IF INSTR(LN$,J$(J))<>0 AND INSTR(LEFT$(LN$,4),"D
                             ATA")=0 THEN LM=0:LL=CL-LM-A-MG:GOSUB 3000:'N
                             EW PAGE
1120                      NEXT J
1130                 'END DO
1140                 'DO FOR K=1 TO KN
1150                      FOR K=1 TO KN
1160                      IF INSTR(LN$,K$(K))<>0 AND INSTR(LEFT$(LN$,4),"D
                             ATA")=0 THEN LM=LM-DM:IF LM<0 THEN LM=0:'DE-I
                             NDENT
1162                      LL=CL-LM-A-MG
1170                      NEXT K
1180                 'END DO
1190                 GOSUB 4000:'PRINT LINE
1200                 'DO FOR M=1 TO MN
1210                      FOR M=1 TO MN
1220                      IF INSTR(LN$,M$(M))<>0 AND INSTR(LEFT$(LN$,4),"D
                             ATA")=0 THEN LM=LM+DM:IF LM<0 THEN LM=0:'INDE
                             NT
1222                      LL=CL-LM-A-MG
1230                      NEXT M
1240                 'END DO
```

```
1241                    GOTO 1249
1242               'ELSE
1244                    PRINT USING A$;NL;:PRINT LN$
1249               'END IF
1250               IF (NOT((EOF(1))OR(NL>=YL)))THEN 1030
1260          'END DO
1270          CLOSE
1280          GOSUB 8000:'FOOT
1998          RETURN
1999     'END SBR
2000     'SBR INITIALIZE
2010          MG$=STRING$(5,32):MG=LEN(MG$)
2020          A$="###### ":G=LEN(A$)
2030          PL=57
2040          LM=0
2050          CL=79
2060          SL=0.5
2070          LI=CL-LM-4
2090          LN=0
2100          DM=5
2200          GOSUB 9000:'GET HEADING
2210          GOSUB 5000:'GET NEW PAGE CRITERIA
2220          GOSUB 6000:'GET DE-INDENT CRITERIA
2230          GOSUB 7000:'GET INDENT CRITERIA
2240          IF XI=0 THEN GOSUB 3000:'NEW PAGE
2998          RETURN
2999     'END SBR
3000     'SBR NEW PAGE
3010          IF PN>=0 GOSUB 8000:'FOOT
3020          PN=PN+1
3030          LN=0
3040          LPRINT H$(0)
3050          'DO FOR H=1 TO HH
3060               FOR H=1 TO HH
```

AD-A127 875 UNIVERSAL MICROCOMPUTER INTERFACE FOR DATA ACQUISITION (U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA W B BROWN MAR 83 2/2

UNCLASSIFIED F/G 9/2 NL





```
3070            LPRINT H$(H)
3080            LN=LN+1
3090            NEXT H
3100        'END DO
3102        LPRINT TAB(MG) TITLE$:LPRINT:LN=LN+2
3998        RETURN
3999 'END DO

4000 'SBR PRINT LINE
4010        PRINT USING A$;NL;
4020        PRINT LN$
4030        LPRINT MG$;
4040        LPRINT USING A$;NL;
4050        'IF LEN(LN$)<LL)) THEN
4060            IF(NOT(LEN(LN$)<LL))THEN 4110
4070            LPRINT STRING$(LM,32);LN$
4080            LN=LN+1
4090            IF LN>=PL GOSUB 3000:'NEW PAGE
4100            GOTO 4250
4110        'ELSE
4120            EF=0
4130            LPRINT STRING$(LM,32);LEFT$(LN$,LL)
4140            LN=LN+1
4150            IF LN>=PL GOSUB 3000:'NEW PAGE
4160            LN$=RIGHT$(LN$,LEN(LN$)-LL)
4165            LM=LM+3:LL=CL-LM-A-MG
4170            'DO UNTIL EF=1
4180                LPRINT MG$;STRING$(A,32);
4190                LPRINT STRING$(LM,32);LEFT$(LN$,LL)
4200                LN=LN+1
4210                IF LN>=PL GOSUB 3000:'NEW PAGE
4220                IF LEN(LN$)<LL THEN EF=1 ELSE LN$=RIGHT$(LN$,LEN(LN$)
                        -LL)
4230                IF(NOT(EF=1))THEN 4170
4240            'END DO
```

```
4245            LM=LM-3:LL=CI-LN-A-M6
4250        'END IF
4998        RETURN
4999  'END SBR
5000  'SBR GET NEW PAGE CRITERIA
5010        RESTORE
5020        T1$="NEW PAGE CRITERIA"
5030        'DO UNTIL T$=T1$
5040            READ T$
5050            IF(NOT(T$=T1$))THEN 5030
5060        'END DO
5070        JN=0
5080        T1$="END NEW PAGE CRITERIA"
5090        'DO UNTIL T$=T1$
5100            JN=JN+1
5110            READ J$(JN)
5120            IF(NOT(J$(JN)=T1$))THEN 5090
5130        'END DO
5140        JN=JN-1
5800        DATA"NEW PAGE CRITERIA"
5810        DATA"'SBR"
5820        DATA"'ALG"
5899        DATA"END NEW PAGE CRITERIA"
5998        RETURN
5999  'END SBR
6000  'SBR GET DE-INDENT CRITERIA
6010        RESTORE
6020        T1$="DE-INDENT CRITERIA"
6030        'DO UNTIL T$=T1$
6040            READ T$
6050            IF(NOT(T$=T1$))THEN 6030
6060        'END DO
6070        KN=0
6080        T1$="END DE-INDENT CRITERIA"
```

```
6090        'DO UNTIL K$(KN)=T1$
6100              KN=KN+1
6110              READ K$(KN)
6120              IF(NOT(K$(KN)=T1$))THEN 6090
6130        'END DO
6140        KN=KN-1
6800        DATA"DE-INDENT CRITERIA"
6810        DATA"'ELSE"
6820        DATA"'END"
6830        DATA"'CASE:"
6899        DATA"END DE-INDENT CRITERIA"
6998        RETURN
6999 'END SBR

7000 'SBR GET INDENT CRITERIA
7010        RESTORE
7020        T1$="INDENT CRITERIA"
7030        'DO UNTIL T$=T1$
7040              READ T$
7050              IF(NOT(T$=T1$))THEN 7030
7060        'END DO
7070        MN=0
7080        T1$="END INDENT CRITERIA"
7090        'DO UNTIL  M$(MN)=T1$
7100              MN=MN+1
7110              READ M$(MN)
7120              IF(NOT(M$(MN)=T1$))THEN 7090
7130        'END DO
7140        MN=MN-1
7800        DATA"INDENT CRITERIA"
7810        DATA"'SUB"
7820        DATA"'ALG"
7830        DATA"'DO"
7840        DATA"'LOOP"
```

```
7850        DATA"'IF"
7860        DATA"'ELSE"
7899        DATA"END INDENT CRITERIA"
7998        RETURN
7999 'END SBR

8000 'SBR FOOT
8010        'DO WHILE LN<=PL
8020              IF (NOT(LN<=PL)) THEN 8060
8030              LPRINT
8040              LN=LN+1
8050              GOTO 8010
8060        'END DO
8070        LPRINT
8080        CP$="-"+STR$(PN)+" -"
8090        GOSUB 10000:'CENTER PRINT
8998        RETURN
8999 'END SBR

9000 'SBR GET H$(/H$(HN),HN)
9010        H$(0)=CHR$(140)
9020        'DO FOR T=1 TO 6
9030              FOR T=1 TO 6
9040              H$(T)=" "
9050              NEXT T
9060        'END DO
9070        HN=6
9998        RETURN
9999 'END SBR
10000 'SBR CENTER PRINT(CP$,MG$,CL/)
10010       T=LEN(CP$)
10020       T1=(CL-T)/2
10030       T$=STRING$(T1,32)+CP$
10040       LPRINT MG$;T$
10998       RETURN
10999 'END SBR
```

```
 1 DATA"*****************************************************"
 2 DATA"***                                                 ***"
 3 DATA"***            UID SUBROUTINE LIBRARY               ***"
 4 DATA"***                                                 ***"
 5 DATA"***            BY LCDR W. B. BROWN, JR              ***"
 6 DATA"***                 200CT82/REV 1.0                 ***
 7 DATA"***                                                 ***"
 8 DATA"*****************************************************"
 9 DATA" "
10 CLS:FOR T=1 TO 9:READ T$:PRINT T$:NEXT:PRINT"PRESS ENTER TO CONTINUE
   "
11 GOSUB 32100:'CR
12 CLS
13 PRINT"RUNNING"
20 CLEAR 5000
22 DIM M$(15,2)

200 'SBR MENU DRIVER
202       'LOOP
204       PRINT
206       INPUT"COMMAND";C$
208       IF C$="END" THEN 297:'QUIT
210       T=0:T$="END"
212       'DO UNTIL C$=M$(T,1) OR M$(T,1)=T$
214             T=T+1
216             IF(NOT((C$=M$(T,1))OR(M$(T,1)=T$)))THEN 212
218       'END DO
220       'IF C$=M$(T,1)) THEN
222             IF(NOT(C$=M$(T,1)))THEN 292
224             TP=VAL(M$(T,2))
226             IF TP>0 AND TP<10 THEN ON IF(TP=1) OR(TP=2) OR(TP=3) OR(TP=4) OR(TP=5) O
                R(TP=6) OR(TP=7) OR(TP=8) OR(TP=9) THEN 290
```

```
228             IF TP>10 AND TP<=20 THEN ON TP-10 GOSUB 11000,12000,13000,
                   14000,15000,16000,17000,18000,19000,20000:GOTO 290
230             IF TP>20 AND TP<=30 THEN ON TP-20 GOSUB 21000,22000,23000,
                   24000,25000,26000,27000,28000,29000,30000:GOTO 290
232             IF TP>30 AND TP<=40 THEN ON TP-30 GOSUB 31000,32000,33000,
                   34000,35000,36000,37000,38000,39000,40000:GOTO 290
234             IF TP>40 AND TP<=50 THEN ON TP-40 GOSUB 41000,42000,43000,
                   44000,45000,46000,47000,48000,49000,50000:GOTO 290
236             IF TP>50 AND TP<=60 THEN ON TP-50 GOSUB 51000,52000,53000,
                   54000,55000,56000,57000,58000,59000,60000
290             GOTO 295
292         'ELSE
294             PRINT"INVALID COMMAND"
295         'END IF
296         GOTO 202
297  'END LOOP
298  RETURN
299  'END SBR

31100  'SBR READ SCANIVALVE TRANSDUCER
31102         TT=0
31104         'DO FOR T=1 TO SN
31105               FOR T=1 TO SN
31106               OUT 0,SC
31108               TT=TT+INP(0)
31110               NEXT T
31112         'END DO
31114         P=TT/SN
31120         RETURN
31122         REM
31124         REM ***************************************************
31126         REM
31128         REM   RETURNS AVERAGE VALUE OF A/D CHANNEL SC IN P
```

```
31130       REM
31132       REM    AUTO SCALING OF P TO MEASURED UNITS
31134       REM
31136       REM    AVERAGES OVER SN READINGS
31138       REM
31140       REM    SCANIVALVE MUST BE SET TO ADDRESS
31142       REM
31144       REM                 *** VARIABLES USED ***
31146       REM
31148       REM        P = A/D PARAMETER
31150       REM
31152       REM        SN = NUMBER OF READINGS TO AVERAGE
31154       REM
31156       REM        SC = A/D CHANNEL
31158       REM
31199 'END SBR
```



```
31200 'SBR READ SCANIVALVE ADDRESS
31202       T=INP(1):T=NOT(T)
31204       SV=T AND 1
31206       SV=SV+(T AND 2)
31208       SV=SV+(T AND 4)
31210       SV=SV+(T AND 8)
31212       SV=SV+10*((T AND 16)/16)
31214       SV=SV+20*((T AND 32)/32)
31216       SV=SV+40*((T AND 64)/64)
31218       RETURN
31280       REM
31281       REM ************************************************************
31282       REM
31283       REM    RETURNS DECIMAL ADDRESS OF SCANIVALVE IN SV
31284       REM
31285       REM    TO D ADDRESS PIN(S) OF SCANIVALVE CONTROLLER TO
31286       REM    DIGITAL INPUT PORT BEGINNING LSB IN BIT 0
```

```
31287       REM
31288       REM          *** VARIABLES USED ***
31289       REM
31290       REM      SV = SCANIVALVE ADDRESS(DECIMAL)
31299 'END SBR

31300 'SBR ADVANCE SCANIVALVE TO ADDRESS
31302       GOSUB 31200:'READ SCANIVALVE
31304       'IF SA<>SV THEN
31306             IF(NOT(SA<>SV))THEN 31324
31308             IF SA<SV GOSUB 31600:'HOME SCANIVALVE
31310             GOSUB 31200:'READ SCANIVALVE
31312             'DO WHILE SA<>SV
31314                   IF(NOT(SA<>SV))THEN 31322
31316                   GOSUB 31700:'STEP SCANIVALVE
31318                   GOSUB 31200:'READ SCANIVALVE
31320                   GOTO 31312
31322             'END DO
31324       'END IF
31326       RETURN
31380       REM
31381       REM ******************************************************
31382       REM
31383       REM   ADVANCES SCANIVALVE TO ADDRESS SA
31384       REM
31385       REM          *** VARIABLES USED ***
31386       REM
31387       REM      SV = SCANIVALEVE ADDRESS (DECIMAL)
31388       REM
31389       REM      SA = TARGET ADDRESS (DECIMAL)
31390       REM
31399 'END SBR
```

```
31400 'SBR ADVANCE SCANIVALVE TO PORT
31402       SA=SP(SP)
31404       GOSUB 31300:'ADVANCE SCANIVALVE TO ADDRESS
31406       RETURN
31480       REM
31481       REM ********************************************************
31482       REM
31483       REM     ADVANCES SCANIVALVE TO TEST SECTION TAP PORT
31484       REM
31485       REM     ARRAY SP CROSS REFERENCES SCANIVALVE ADDRESS
31486       REM     WITH TEST PORT
31487       REM
31488       REM           *** VARIABLES USED ***
31489       REM
31490       REM      SP = TARGET PORT NUMBER (DECIMAL)
31491       REM
31492       REM      SP(SP) = PORT/SCANIVALVE ADDRESS CROSS REFERENCE
31493       REM
31499 'END SBR
31500 'SBR CALIBRATE SCANIVALVE TRANSDUCER
31501       CLS:PRINT"WAIT"
31502       'DO UNTIL FR$="H"
31504             SP=SL
31506             GOSUB 31400:'ADVANCE SCANIVALVE TO PORT
31507             CLS:GOSUB 3000:'ADDRESS
31508             PRINT
31510             PRINT"ADJUST TRANSDUCER BIAS ON VOLTMETER TO MINIMUM"
31512             PRINT"LOWER BOUNDARY OF 1 PSIG RANGE."
31514             PRINT
31516             PRINT"PRESS ENTER TO CONTINUE"
31518             GOSUB 32100:'CR
31519             CLS:PRINT"WAIT"
31520             SP=SU
```

```
31522             GOSUB 31400:'ADVANCE SCANIVALVE TO PORT
31523             CLS
31524             GOSUB 3000:'ADDRESS
31525             PRINT
31526             PRINT"ADJUST TRANSDUCER GAIN ON VOLTMETER TO WITHIN"
31528             PRINT"UPPER BOUNDARY OF LINEAR RANGE"
31530             PRINT
31532             PRINT"PRESS ENTER TO CONTINUE"
31534             GOSUB 32100:'CR
31536             PRINT
31538             V$="WANT TO RECHECK"
31539             CLS
  540             GOSUB 32200:'Y/N
 1542             IF(NOT(KB$="N"))THEN 31502
31544        'END DO
31546        RETURN
  580        REM
3...1        REM ******************************************************
31582        REM
31583        REM   ALLOWS PAUSE IN PROGRAM EXECUTION TO SET GAIN AND
31584        REM   BIAS OF CONDITIONING AMPLIFIER BETWEEN PORTS
31585        REM   SL AND SU
31586        REM
31587        REM   ALLOWS FOR MULTIPLE CALIBRATION CHECK
31588        REM
31589        REM   REQUIRES DESIGNATION OF SAMPLE PORTS PRIOR TO
31590        REM   EXECUTION
31591        REM
31592        REM           *** VARIABLES USED ***
31593        REM
31594        REM     SL,SU = SAMPLE PORTS
31599 'END DO
```

```
31600 'SBR HOME SCANIVALVE
31602       VI=SH
31604       GOSUB 32300:'DECIMAL TO BIT ENCODE
31606       B=VO
31608       OUT 1,B
31610       FOR TD=1 TO 50:NEXT:'DELAY
31612       OUT 1,0
31614       'DO UNTIL SV=48
31616            GOSUB 31200:'READ SCANIVALVE
31618            IF(NOT(SV=48))THEN 31614
31620       'END DO
31621       PRINT"SCANIVALVE HOME":GOSUB 3000:'ADDRESS
31622       RETURN
31680       REM
31681       REM ********************************************************
31682       REM
31683       REM   HOMES SCANIVALVE TO ADDRESS 48
31684       REM
31685       REM   SCANIVALVE HOME CONTROL CONNECTED TO RELAY SH
31686       REM
31687       REM            *** VARIABLES USED ***
31688       REM
31689       REM      SH = HOME CONTROL BIT (DECIMAL)
31690       REM
31691       REM      B = BIT POSITION (DECIMAL ENCODED BINARY)
31692       REM
31693       REM      SA = SCANIVALVE ADDRESS
31699 'END SBR
31700 'SBR STEP SCANIVALVE
31702       GOSUB 31200:'READ SCANIVALVE
31704       ST=SV+1
31706       IF ST>48 THEN ST=ST-48
31708       VI=SS
```

```
31710       GOSUB 32300:'DECIMAL TO BIT ENCODE
31712       B=V0
31714       'DO UNTIL SV=ST
31716            OUT 1,B
31718            FOR TD=1 TO 2:NEXT:'DELAY
31720            OUT 1,0
31724            GOSUB 31200:'READ SCANIVALVE ADDRESS
31726            IF(NOT(SV=ST))THEN 31714
31728       'END DO
31730       RETURN
31780       REM
31781       REM ***************************************************
31782       REM
31783       REM    ADVANCES SCANIVALVE ADDRESS BY ONE
31784       REM
31785       REM    SCANIVALVE STEP CONTROL CONNECTED TO RELAY 55
31786       REM
31787       REM          *** VARIABLES USED ***
31788       REM
31789       REM     SV = SCANIVALVE ADDRESS (DECIMAL)
31790       REM     ST = TARGET ADDRESS (DECIMAL)
31791       REM     SS = SCANIVALVE STEP CONTROL BIT
31792       REM     B = BIT POSITION (DECIMAL ENCODED BINARY)
31793       REM
31799 'END SBR

31800 'SBR SCALE A/D PORT
31802       V0=V1*M(SC)+B(SC)
31804       RETURN
31880       REM
31881       REM ***************************************************
31882       REM
31883       REM   CONVERTS NON DIMENSIONAL A/D UNITS TO UNITS
31884       REM
```

```
31885       REM . CHANNEL MUST BE CALIBRATED AND SCALED
31886       REM   BEFORE EXECUTION
31887       REM
31888       REM            *** VARIABLES USED ***
31889       REM
31890       REM      VT = NON DIMENSIONAL INPUT
31891       REM      SC = A/D CHANNEL NUMBER
31892       REM      VI = NON DIMENSIONAL INPUT
31893       REM      VO = DIMENSIONAL OUTPUT
31894       REM
31899 'END SBR

31900 'SBR SCALE SCANIVALVE TRANSDUCER
31902       SP=50
31904       GOSUB 31400:'ADVANCE SCANIVALVE TO PORT
31906       PRINT
31908       PRINT"ENTER READING OF PORT";SP
31910       INPUT XO
31912       GOSUB 31100:'READ SCANIVALVE TRANSDUCER
31914       XO=P
31916       SP=51
31918       GOSUB 31400:'ADVANCE SCANIVALVE TO PORT
31920       PRINT
31922       PRINT"ENTER READING OF PORT";SP
31924       INPUT YI
31926       GOSUB 31100:'READ SCANIVALVE TRANSDUCER
31928       PRINT
31930       INPUT"ENTER UNITS";SU$(SC)
31932       X1=P
31934       M(SC)=(YI-YO)/(X1-XO)
31936       B(SC)=(YO-(XO*M(SC)))-(YI-(X1*M(SC)))
31938       RETURN
31980       REM
```

```
31981       REM ************************************************************
31982       REM
31983       REM   RETURNS LINEAR SCALE CONVERSION FACTORS FOR A/D
31984       REM   CHANNEL SC:   Y = M*X + B
31985       REM
31986       REM   REQUIRES DESIGNATION OF SAMPLE PORTS BEFORE
31987       REM   EXECUTION
31988       REM
31989       REM            *** VARIABLES USED ***
31990       REM
31991       REM       SC = A/D CHANNEL
31992       REM       SL,SU = SAMPLE PORTS
31993       REM       M(SC) = CONVERSION SLOPE
31994       REM       B(SC) = CONVERSION BIAS
31995       REM
31999 'END SBR

32100 'SBR CR
32102       KB$=INKEY$:IF KB$<>CHR$(13) THEN 32102
32104       RETURN
32180       REM
32181       REM ************************************************************
32182       REM
32183       REM    LOOPS UNTIL <ENTER> KEY IS DEPRESSED
32184       REM
32199 'END SBR

32200 'SBR Y/N
32202       PRINT V$;" (Y/N) ";
32204       'DO UNTIL KB$="Y" OR KB$="N"
32206             KB$=INKEY$:IF KB$="" THEN 32206
32208             IF(NOT((KB$="Y")OR(KB$="N"))) THEN 32204
32210       'END DO
32212       PRINT KB$
32214       RETURN
```

```
32280       REM
32281       REM ************************************************************
32282       REM
32283       REM    LOOPS UNTIL "Y" OR "N" IS DEPRESSED INDICATING
32284       REM    A "YES" OR "NO" RESPONSE
32285       REM
32286       REM              *** VARIABLES USED ***
32287       REM
32288       REM       I$* = RESPONSE
32289       REM
32299 'END SBR

32300 'SBR DECIMAL TO BIT ENCODE
32302       V0=2^V1
32304       RETURN
32380       REM
32381       REM ************************************************************
32382       REM
32383       REM     RETURNS BINARY ENCODED DECIMAL FOR BIT ENCODING
32384       REM     OF RELAY OR DIGITAL PORTS
32385       REM
32399 'END SBR

32400 'SBR CONVERT BCD TO DEC IMAG
32402       V=0
32404       'DO FOR I=V0 TO V0+4
32406             FOR I=V0 TO V0+4
32408             V0=V0+(V1 AND (2^I-1))
32410             NEXT I
32412       'END DO
32414       RETURN
32480       REM
32481       REM************************************************************
32482       REM
```

```
32483       REM         CONVERTS BCD NUMBER VI TO DECIMAL NUMBER VO.
32484       REM         VB = BIT POSITION LSB OF VI
32485       REM
32486       REM              *** VARIABLES USED ***
32487       REM
32488       REM              VI = BCD INPUT PARAMETER
32489       REM              VO = DECIMAL OUTPUT
32490       REM              VB = LSB OF VI
32491       REM
32499 'END SBR

32500 'SBR UNMASK BIT PATTERNS
32502       'DO FOR T=0 TO 7
32504            FOR T=0 TO 7
32506            VO(T)=(VI AND (2^T))/(2^T)
32508            NEXT T
32510       'END DO
32512       RETURN
32590       REM
32591       REM*************************************************************
32592       REM
32593       REM       RETURNS BIT STATUS OF VI IN VO(0) TO VO(7)
32594       REM
32595       REM              *** VARIABLES USED ***
32596       REM
32597       REM         VI = DECIMAL INPUT
32598       REM         VO(0) TO VO(7) = BIT OUTPUT
32599 'END SBR

32600 'SBR MASK BIT PATTERNS
32602       VO=0
32604       'DO FOR T=0 TO 7
32606            FOR T=0 TO 7
32608            VO=VO+(VI(T) AND (2^T))
32610            NEXT T
```

```
32612       'END DO
32614       RETURN

32680       REM
32681       REM*************************************************************
32682       REM
32683       REM      RETURNS DECIMAL NUMBER VO ACCORDING TO BINARY
32684       REM      WEIGHTED VALUES OF VI(1) TO VI(7)
32685       REM
32686       REM           *** VARIABLES USED ***
32687       REM
32688       REM        VI(1) TO VI(7) = BIT INPUTS
32689       REM        VO = DECIMAL OUTPUT
32690       REM
32699 'END SUB
```

APPENDIX G

SYSTEM CALIBRATION RESULTS

The following pages contain the digital truth tables, calibration tables for the D/A and A/D ports, and associated plots of these data.

In the D/A tables, the output port is represented in the first two columns labeled ARG and VOLTS. The remaining eight columns are not used. ARG is the non-dimensional integer argument sent to the D/A output port and VOLTS is the resulting analog output voltage in volts at the METER SELECT jack position 8 measured to four places with a digital volt meter.

In the A/D tables, the D/A output port was used to drive each of the A/D input channels through an external conditioning amplifier biased to sweep through the range of the channels. ARG is the non-dimensional integer argument to the D/A port, the VOLTS column is the amplified input to the A/D channels in volts, and the CH 0 to CH 7 columns are the returned non-dimensional integer parameters from the respective A/D channel.

DECIMAL/BINARY TRUTH TABLE

0	00000000	1	10000000	2	01000000	3	11000000
4	00100000	5	10100000	6	01100000	7	11100000
8	00010000	9	10010000	10	01010000	11	11010000
12	00110000	13	10110000	14	01110000	15	11110000
16	00001000	17	10001000	18	01001000	19	11001000
20	00101000	21	10101000	22	01101000	23	11101000
24	00011000	25	10011000	26	01011000	27	11011000
28	00111000	29	10111000	30	01111000	31	11111000
32	00000100	33	10000100	34	01000100	35	11000100
36	00100100	37	10100100	38	01100100	39	11100100
40	00010100	41	10010100	42	01010100	43	11010100
44	00110100	45	10110100	46	01110100	47	11110100
48	00001100	49	10001100	50	01001100	51	11001100
52	00101100	53	10101100	54	01101100	55	11101100
56	00011100	57	10011100	58	01011100	59	11011100
60	00111100	61	10111100	62	01111100	63	11111100
64	00000010	65	10000010	66	01000010	67	11000010
68	00100010	69	10100010	70	01100010	71	11100010
72	00010010	73	10010010	74	01010010	75	11010010
76	00110010	77	10110010	78	01110010	79	11110010
80	00001010	81	10001010	82	01001010	83	11001010
84	00101010	85	10101010	86	01101010	87	11101010
88	00011010	89	10011010	90	01011010	91	11011010
92	00111010	93	10111010	94	01111010	95	11111010
96	00000110	97	10000110	98	01000110	99	11000110
100	00100110	101	10100110	102	01100110	103	11100110
104	00010110	105	10010110	106	01010110	107	11010110
108	00110110	109	10110110	110	01110110	111	11110110
112	00001110	113	10001110	114	01001110	115	11001110
116	00101110	117	10101110	118	01101110	119	11101110
120	00011110	121	10011110	122	01011110	123	11011110
124	00111110	125	10111110	126	01111110	127	11111110
128	00000001	129	10000001	130	01000001	131	11000001

132	00100001	133	10100001	134	01100001	135	11100001
136	00010001	137	10010001	138	01010001	139	11010001
140	00110001	141	10110001	142	01110001	143	11110001
144	00001001	145	10001001	146	01001001	147	11001001
148	00101001	149	10101001	150	01101001	151	11101001
152	00011001	153	10011001	154	01011001	155	11011001
156	00111001	157	10111001	158	01111001	159	11111001
160	00000101	161	10000101	162	01000101	163	11000101
164	00100101	165	10100101	166	01100101	167	11100101
168	00010101	169	10010101	170	01010101	171	11010101
172	00110101	173	10110101	174	01110101	175	11110101
176	00001101	177	10001101	178	01001101	179	11001101
180	00101101	181	10101101	182	01101101	183	11101101
184	00011101	185	10011101	186	01011101	187	11011101
188	00111101	189	10111101	190	01111101	191	11111101
192	00000011	193	10000011	194	01000011	195	11000011
196	00100011	197	10100011	198	01100011	199	11100011
200	00010011	201	10010011	202	01010011	203	11010011
204	00110011	205	10110011	206	01110011	207	11110011
208	00001011	209	10001011	210	01001011	211	11001011
212	00101011	213	10101011	214	01101011	215	11101011
216	00011011	217	10011011	218	01011011	219	11011011
220	00111011	221	10111011	222	01111011	223	11111011
224	00000111	225	10000111	226	01000111	227	11000111
228	00100111	229	10100111	230	01100111	231	11100111
232	00010111	233	10010111	234	01010111	235	11010111
236	00110111	237	10110111	238	01110111	239	11110111
240	00001111	241	10001111	242	01001111	243	11001111
244	00101111	245	10101111	246	01101111	247	11101111
248	00011111	249	10011111	250	01011111	251	11011111
252	00111111	253	10111111	254	01111111	255	11111111

PAGE 1

UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	D/A OUTPUT PORT CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
0	0.025	0	0	0	0	0	0	0	0
1	0.044	0	0	0	0	0	0	0	0
2	0.063	0	0	0	0	0	0	0	0
3	0.082	0	0	0	0	0	0	0	0
4	0.097	0	0	0	0	0	0	0	0
5	0.116	0	0	0	0	0	0	0	0
6	0.135	0	0	0	0	0	0	0	0
7	0.154	0	0	0	0	0	0	0	0
8	0.172	0	0	0	0	0	0	0	0
9	0.191	0	0	0	0	0	0	0	0
10	0.210	0	0	0	0	0	0	0	0
11	0.229	0	0	0	0	0	0	0	0
12	0.244	0	0	0	0	0	0	0	0
13	0.263	0	0	0	0	0	0	0	0
14	0.282	0	0	0	0	0	0	0	0
15	0.301	0	0	0	0	0	0	0	0
16	0.302	0	0	0	0	0	0	0	0
17	0.321	0	0	0	0	0	0	0	0
18	0.340	0	0	0	0	0	0	0	0
19	0.359	0	0	0	0	0	0	0	0
20	0.374	0	0	0	0	0	0	0	0
21	0.393	0	0	0	0	0	0	0	0
22	0.412	0	0	0	0	0	0	0	0
23	0.431	0	0	0	0	0	0	0	0
24	0.449	0	0	0	0	0	0	0	0
25	0.468	0	0	0	0	0	0	0	0
26	0.487	0	0	0	0	0	0	0	0
27	0.505	0	0	0	0	0	0	0	0
28	0.521	0	0	0	0	0	0	0	0
29	0.540	0	0	0	0	0	0	0	0
30	0.559	0	0	0	0	0	0	0	0
31	0.578	0	0	0	0	0	0	0	0
32	0.607	0	0	0	0	0	0	0	0
33	0.626	0	0	0	0	0	0	0	0
34	0.645	0	0	0	0	0	0	0	0
35	0.664	0	0	0	0	0	0	0	0
36	0.679	0	0	0	0	0	0	0	0
37	0.698	0	0	0	0	0	0	0	0
38	0.717	0	0	0	0	0	0	0	0
39	0.736	0	0	0	0	0	0	0	0
40	0.754	0	0	0	0	0	0	0	0
41	0.773	0	0	0	0	0	0	0	0
42	0.792	0	0	0	0	0	0	0	0
43	0.811	0	0	0	0	0	0	0	0
44	0.826	0	0	0	0	0	0	0	0
45	0.845	0	0	0	0	0	0	0	0

PAGE 2
UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
46	0.864	0	0	0	0	0	0	0	0
47	0.883	0	0	0	0	0	0	0	0
48	0.884	0	0	0	0	0	0	0	0
49	0.903	0	0	0	0	0	0	0	0
50	0.922	0	0	0	0	0	0	0	0
51	0.941	0	0	0	0	0	0	0	0
52	0.956	0	0	0	0	0	0	0	0
53	0.975	0	0	0	0	0	0	0	0
54	0.994	0	0	0	0	0	0	0	0
55	1.013	0	0	0	0	0	0	0	0
56	1.031	0	0	0	0	0	0	0	0
57	1.050	0	0	0	0	0	0	0	0
58	1.069	0	0	0	0	0	0	0	0
59	1.088	0	0	0	0	0	0	0	0
60	1.103	0	0	0	0	0	0	0	0
61	1.122	0	0	0	0	0	0	0	0
62	1.141	0	0	0	0	0	0	0	0
63	1.160	0	0	0	0	0	0	0	0
64	1.193	0	0	0	0	0	0	0	0
65	1.212	0	0	0	0	0	0	0	0
66	1.231	0	0	0	0	0	0	0	0
67	1.250	0	0	0	0	0	0	0	0
68	1.265	0	0	0	0	0	0	0	0
69	1.284	0	0	0	0	0	0	0	0
70	1.303	0	0	0	0	0	0	0	0
71	1.322	0	0	0	0	0	0	0	0
72	1.340	0	0	0	0	0	0	0	0
73	1.359	0	0	0	0	0	0	0	0
74	1.378	0	0	0	0	0	0	0	0
75	1.396	0	0	0	0	0	0	0	0
76	1.412	0	0	0	0	0	0	0	0
77	1.531	0	0	0	0	0	0	0	0
78	1.450	0	0	0	0	0	0	0	0
79	1.469	0	0	0	0	0	0	0	0
80	1.470	0	0	0	0	0	0	0	0
81	1.489	0	0	0	0	0	0	0	0
82	1.508	0	0	0	0	0	0	0	0
83	1.527	0	0	0	0	0	0	0	0
84	1.543	0	0	0	0	0	0	0	0
85	1.562	0	0	0	0	0	0	0	0
86	1.581	0	0	0	0	0	0	0	0
87	1.600	0	0	0	0	0	0	0	0
88	1.618	0	0	0	0	0	0	0	0
89	1.637	0	0	0	0	0	0	0	0
90	1.656	0	0	0	0	0	0	0	0
91	1.674	0	0	0	0	0	0	0	0

PAGE 3
UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
92	1.690	0	0	0	0	0	0	0	0
93	1.709	0	0	0	0	0	0	0	0
94	1.728	0	0	0	0	0	0	0	0
95	1.746	0	0	0	0	0	0	0	0
96	1.776	0	0	0	0	0	0	0	0
97	1.795	0	0	0	0	0	0	0	0
98	1.814	0	0	0	0	0	0	0	0
99	1.833	0	0	0	0	0	0	0	0
100	1.848	0	0	0	0	0	0	0	0
101	1.867	0	0	0	0	0	0	0	0
102	1.886	0	0	0	0	0	0	0	0
103	1.905	0	0	0	0	0	0	0	0
104	1.923	0	0	0	0	0	0	0	0
105	1.942	0	0	0	0	0	0	0	0
106	1.961	0	0	0	0	0	0	0	0
107	1.980	0	0	0	0	0	0	0	0
108	1.995	0	0	0	0	0	0	0	0
109	2.014	0	0	0	0	0	0	0	0
110	2.033	0	0	0	0	0	0	0	0
111	2.052	0	0	0	0	0	0	0	0
112	2.054	0	0	0	0	0	0	0	0
113	2.073	0	0	0	0	0	0	0	0
114	2.092	0	0	0	0	0	0	0	0
115	2.110	0	0	0	0	0	0	0	0
116	2.126	0	0	0	0	0	0	0	0
117	2.145	0	0	0	0	0	0	0	0
118	2.164	0	0	0	0	0	0	0	0
119	2.183	0	0	0	0	0	0	0	0
120	2.201	0	0	0	0	0	0	0	0
121	2.220	0	0	0	0	0	0	0	0
122	2.239	0	0	0	0	0	0	0	0
123	2.258	0	0	0	0	0	0	0	0
124	2.273	0	0	0	0	0	0	0	0
125	2.292	0	0	0	0	0	0	0	0
126	2.311	0	0	0	0	0	0	0	0
127	2.330	0	0	0	0	0	0	0	0
128	2.360	0	0	0	0	0	0	0	0
129	2.379	0	0	0	0	0	0	0	0
130	2.398	0	0	0	0	0	0	0	0
131	2.416	0	0	0	0	0	0	0	0
132	2.432	0	0	0	0	0	0	0	0
133	2.451	0	0	0	0	0	0	0	0
134	2.470	0	0	0	0	0	0	0	0
135	2.489	0	0	0	0	0	0	0	0
136	2.507	0	0	0	0	0	0	0	0
137	2.526	0	0	0	0	0	0	0	0

PAGE 4
UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
138	2.545	0	0	0	0	0	0	0	0
139	2.564	0	0	0	0	0	0	0	0
140	2.579	0	0	0	0	0	0	0	0
141	2.598	0	0	0	0	0	0	0	0
142	2.617	0	0	0	0	0	0	0	0
143	2.636	0	0	0	0	0	0	0	0
144	2.637	0	0	0	0	0	0	0	0
145	2.656	0	0	0	0	0	0	0	0
146	2.675	0	0	0	0	0	0	0	0
147	2.694	0	0	0	0	0	0	0	0
148	2.710	0	0	0	0	0	0	0	0
149	2.728	0	0	0	0	0	0	0	0
150	2.747	0	0	0	0	0	0	0	0
151	2.766	0	0	0	0	0	0	0	0
152	2.785	0	0	0	0	0	0	0	0
153	2.803	0	0	0	0	0	0	0	0
154	2.822	0	0	0	0	0	0	0	0
155	2.841	0	0	0	0	0	0	0	0
156	2.857	0	0	0	0	0	0	0	0
157	2.875	0	0	0	0	0	0	0	0
158	2.894	0	0	0	0	0	0	0	0
159	2.913	0	0	0	0	0	0	0	0
160	2.943	0	0	0	0	0	0	0	0
161	2.962	0	0	0	0	0	0	0	0
162	2.981	0	0	0	0	0	0	0	0
163	3.000	0	0	0	0	0	0	0	0
164	3.015	0	0	0	0	0	0	0	0
165	3.034	0	0	0	0	0	0	0	0
166	3.053	0	0	0	0	0	0	0	0
167	3.072	0	0	0	0	0	0	0	0
168	3.090	0	0	0	0	0	0	0	0
169	3.109	0	0	0	0	0	0	0	0
170	3.128	0	0	0	0	0	0	0	0
171	3.147	0	0	0	0	0	0	0	0
172	3.163	0	0	0	0	0	0	0	0
173	3.182	0	0	0	0	0	0	0	0
174	3.201	0	0	0	0	0	0	0	0
175	3.219	0	0	0	0	0	0	0	0
176	3.221	0	0	0	0	0	0	0	0
177	3.240	0	0	0	0	0	0	0	0
178	3.259	0	0	0	0	0	0	0	0
179	3.278	0	0	0	0	0	0	0	0
180	3.293	0	0	0	0	0	0	0	0
181	3.312	0	0	0	0	0	0	0	0
182	3.331	0	0	0	0	0	0	0	0
183	3.350	0	0	0	0	0	0	0	0

PAGE 5
UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
184	3.368	0	0	0	0	0	0	0	0
185	3.387	0	0	0	0	0	0	0	0
186	3.406	0	0	0	0	0	0	0	0
187	3.425	0	0	0	0	0	0	0	0
188	3.440	0	0	0	0	0	0	0	0
189	3.459	0	0	0	0	0	0	0	0
190	3.478	0	0	0	0	0	0	0	0
191	3.497	0	0	0	0	0	0	0	0
192	3.530	0	0	0	0	0	0	0	0
193	3.549	0	0	0	0	0	0	0	0
194	3.568	0	0	0	0	0	0	0	0
195	3.587	0	0	0	0	0	0	0	0
196	3.602	0	0	0	0	0	0	0	0
197	3.621	0	0	0	0	0	0	0	0
198	3.640	0	0	0	0	0	0	0	0
199	3.659	0	0	0	0	0	0	0	0
200	3.677	0	0	0	0	0	0	0	0
201	3.696	0	0	0	0	0	0	0	0
202	3.715	0	0	0	0	0	0	0	0
203	3.734	0	0	0	0	0	0	0	0
204	3.750	0	0	0	0	0	0	0	0
205	3.769	0	0	0	0	0	0	0	0
206	3.788	0	0	0	0	0	0	0	0
207	3.807	0	0	0	0	0	0	0	0
208	3.808	0	0	0	0	0	0	0	0
209	3.827	0	0	0	0	0	0	0	0
210	3.846	0	0	0	0	0	0	0	0
211	3.865	0	0	0	0	0	0	0	0
212	3.881	0	0	0	0	0	0	0	0
213	3.900	0	0	0	0	0	0	0	0
214	3.918	0	0	0	0	0	0	0	0
215	3.937	0	0	0	0	0	0	0	0
216	3.956	0	0	0	0	0	0	0	0
217	3.974	0	0	0	0	0	0	0	0
218	3.993	0	0	0	0	0	0	0	0
219	4.012	0	0	0	0	0	0	0	0
220	4.028	0	0	0	0	0	0	0	0
221	4.047	0	0	0	0	0	0	0	0
222	4.066	0	0	0	0	0	0	0	0
223	4.085	0	0	0	0	0	0	0	0
224	4.115	0	0	0	0	0	0	0	0
225	4.134	0	0	0	0	0	0	0	0
226	4.153	0	0	0	0	0	0	0	0
227	4.171	0	0	0	0	0	0	0	0
228	4.187	0	0	0	0	0	0	0	0
229	4.206	0	0	0	0	0	0	0	0

PAGE 6
UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
230	4.225	0	0	0	0	0	0	0	0
231	4.243	0	0	0	0	0	0	0	0
232	4.262	0	0	0	0	0	0	0	0
233	4.281	0	0	0	0	0	0	0	0
234	4.300	0	0	0	0	0	0	0	0
235	4.319	0	0	0	0	0	0	0	0
236	4.335	0	0	0	0	0	0	0	0
237	4.354	0	0	0	0	0	0	0	0
238	4.373	0	0	0	0	0	0	0	0
239	4.391	0	0	0	0	0	0	0	0
240	4.393	0	0	0	0	0	0	0	0
241	4.412	0	0	0	0	0	0	0	0
242	4.431	0	0	0	0	0	0	0	0
243	4.450	0	0	0	0	0	0	0	0
244	4.465	0	0	0	0	0	0	0	0
245	4.484	0	0	0	0	0	0	0	0
246	4.503	0	0	0	0	0	0	0	0
247	4.522	0	0	0	0	0	0	0	0
248	4.540	0	0	0	0	0	0	0	0
249	4.559	0	0	0	0	0	0	0	0
250	4.578	0	0	0	0	0	0	0	0
251	4.597	0	0	0	0	0	0	0	0
252	4.612	0	0	0	0	0	0	0	0
253	4.631	0	0	0	0	0	0	0	0
254	4.650	0	0	0	0	0	0	0	0
255	4.669	0	0	0	0	0	0	0	0

PAGE 1
UID A/D & D/A CALIBRATION TABLE A/D INPUT PORT

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
0	0.023	7	1	1	1	2	1	1	1
1	0.043	2	2	2	2	2	2	1	1
2	0.064	3	3	3	3	3	3	2	2
3	0.084	4	4	4	4	4	4	3	3
4	0.101	5	6	5	5	5	5	4	4
5	0.121	6	6	5	6	6	6	5	5
6	0.142	7	7	7	7	7	7	6	6
7	0.162	8	8	8	8	8	8	7	7
8	0.182	9	9	9	9	9	9	8	8
9	0.202	10	10	10	10	10	10	9	9
10	0.223	11	12	11	11	11	11	10	10
11	0.243	12	12	12	12	12	12	11	11
12	0.261	13	13	13	13	13	13	12	12
13	0.281	14	14	14	14	14	14	13	14
14	0.302	15	15	15	15	15	15	14	14
15	0.322	16	16	16	16	16	16	15	15
16	0.324	16	16	16	16	16	16	15	15
17	0.345	17	17	17	17	17	17	16	16
18	0.365	18	19	18	18	18	18	17	17
19	0.386	19	20	19	19	19	19	18	18
20	0.402	20	20	20	20	20	20	19	19
21	0.423	21	21	21	21	21	21	20	20
22	0.443	22	22	22	22	22	22	21	21
23	0.464	23	24	23	23	23	23	22	22
24	0.484	24	24	24	24	24	24	23	23
25	0.504	25	25	25	25	25	25	24	24
26	0.525	26	27	26	26	26	26	25	26
27	0.545	27	27	27	27	27	27	26	26
28	0.562	28	28	28	28	28	28	27	27
29	0.582	29	29	29	29	29	29	28	28
30	0.603	30	30	30	30	30	30	29	29
31	0.623	31	31	31	31	31	31	30	30
32	0.656	33	34	33	33	33	33	32	32
33	0.676	34	34	34	33	34	34	33	34
34	0.696	35	35	35	35	35	35	34	34
35	0.717	36	36	36	36	36	36	35	35
36	0.734	37	37	36	36	36	37	36	36
37	0.754	38	38	37	37	38	38	37	37
38	0.775	38	39	38	38	38	39	38	38
39	0.795	40	40	40	40	40	40	39	39
40	0.815	41	41	41	40	40	41	40	40
41	0.835	42	42	42	41	42	42	41	41
42	0.856	43	43	43	43	43	43	42	42
43	0.877	44	44	44	44	44	44	44	43
44	0.893	45	45	45	44	45	45	44	44
45	0.914	46	46	45	45	45	46	45	45

PAGE 2
UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
46	0.933	47	47	47	47	46	47	46	46
47	0.955	48	48	48	48	48	48	47	47
48	0.957	48	48	48	48	48	48	47	47
49	0.977	49	49	49	49	49	49	48	48
50	0.998	50	50	50	50	50	50	49	49
51	1.018	51	52	51	51	51	51	50	50
52	1.035	52	52	51	51	52	52	51	51
53	1.055	53	53	52	53	52	53	52	52
54	1.076	54	54	53	54	53	54	53	53
55	1.097	55	55	54	55	54	55	54	54
56	1.116	56	56	56	56	56	56	55	55
57	1.137	57	57	57	57	57	57	56	56
58	1.157	58	58	57	58	58	58	57	57
59	1.178	59	59	59	59	59	59	58	58
60	1.195	60	60	59	59	59	60	59	59
61	1.215	61	61	60	60	60	61	60	60
62	1.236	62	62	61	61	61	62	61	61
63	1.256	63	63	62	62	62	63	62	62
64	1.292	65	65	65	65	65	65	64	64
65	1.313	66	66	66	66	66	66	65	65
66	1.333	66	67	67	67	67	67	66	66
67	1.354	68	68	68	68	68	68	67	67
68	1.370	69	69	69	68	69	69	68	68
69	1.391	70	70	69	70	70	70	69	69
70	1.411	71	71	71	71	71	71	70	70
71	1.432	72	72	72	72	72	72	71	71
72	1.452	73	73	73	73	73	73	72	72
73	1.472	74	74	74	74	73	74	73	73
74	1.493	75	75	75	75	75	75	74	74
75	1.513	76	76	76	76	76	76	75	75
76	1.530	77	77	76	77	76	77	76	76
77	1.550	78	78	78	78	78	78	77	77
78	1.571	79	79	79	79	79	79	78	78
79	1.591	80	80	80	80	80	80	79	79
80	1.593	80	80	80	80	80	80	79	79
81	1.614	81	81	81	81	81	81	80	80
82	1.634	82	82	82	82	82	82	82	81
83	1.655	83	83	83	83	83	83	82	82
84	1.672	84	84	84	84	83	84	83	83
85	1.692	85	85	85	85	85	85	84	84
86	1.712	86	86	86	86	85	86	85	85
87	1.733	87	87	87	87	87	87	86	86
88	1.753	88	88	88	88	88	88	87	87
89	1.777	89	89	89	89	89	89	88	88
90	1.794	90	90	90	90	90	90	90	89
91	1.814	91	91	91	91	91	91	90	90

PAGE 3
UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
92	1.831	92	92	92	92	92	92	91	91
93	1.852	93	93	93	93	92	93	92	92
94	1.872	94	94	94	94	94	94	93	93
95	1.892	95	95	95	95	95	95	94	94
96	1.924	96	97	96	96	96	97	96	96
97	1.945	97	98	97	97	97	98	97	97
98	1.966	99	99	98	98	98	99	98	98
99	1.986	99	100	99	99	99	100	99	99
100	2.003	101	101	100	100	100	101	100	100
101	2.023	101	102	101	101	101	102	101	101
102	2.044	103	103	102	102	102	103	102	102
103	2.064	103	104	103	103	103	104	103	103
104	2.084	104	105	104	104	104	105	104	104
105	2.104	106	106	105	105	105	106	105	105
106	2.125	106	107	107	106	106	107	106	106
107	2.145	108	108	107	108	107	108	107	107
108	2.162	108	109	108	109	108	109	108	108
109	2.183	109	110	109	109	109	110	109	109
110	2.203	110	111	110	110	110	111	110	110
111	2.224	111	112	111	111	111	112	111	111
112	2.226	112	112	111	112	111	112	111	111
113	2.246	113	113	112	113	112	113	112	112
114	2.267	113	114	113	114	113	114	113	113
115	2.287	115	115	114	115	114	115	114	114
116	2.304	115	116	115	116	115	116	115	115
117	2.324	117	117	116	117	116	117	116	116
118	2.345	117	118	117	117	117	118	117	117
119	2.365	119	119	118	119	118	119	118	118
120	2.385	120	120	119	120	119	120	119	119
121	2.405	121	121	120	121	120	121	120	120
122	2.426	122	122	121	122	121	122	121	121
123	2.446	123	123	119	123	122	123	122	122
124	2.466	124	124	123	124	123	124	123	123
125	2.483	124	125	124	125	124	125	124	124
126	2.504	125	126	125	126	125	126	125	125
127	2.524	126	127	126	127	126	127	126	126
128	2.557	128	129	128	128	128	129	127	128
129	2.578	129	130	129	130	129	130	129	129
130	2.598	131	131	130	130	130	131	130	130
131	2.618	132	132	131	132	131	132	131	131
132	2.635	132	133	132	132	132	133	131	132
133	2.656	133	134	133	133	134	134	133	133
134	2.676	134	135	134	134	134	135	133	134
135	2.697	135	136	135	135	135	136	135	135
136	2.717	136	137	136	137	136	137	135	136
137	2.737	137	138	137	138	137	138	137	137

PAGE 4
UID A/D & D/A CALIBRATION TABLE

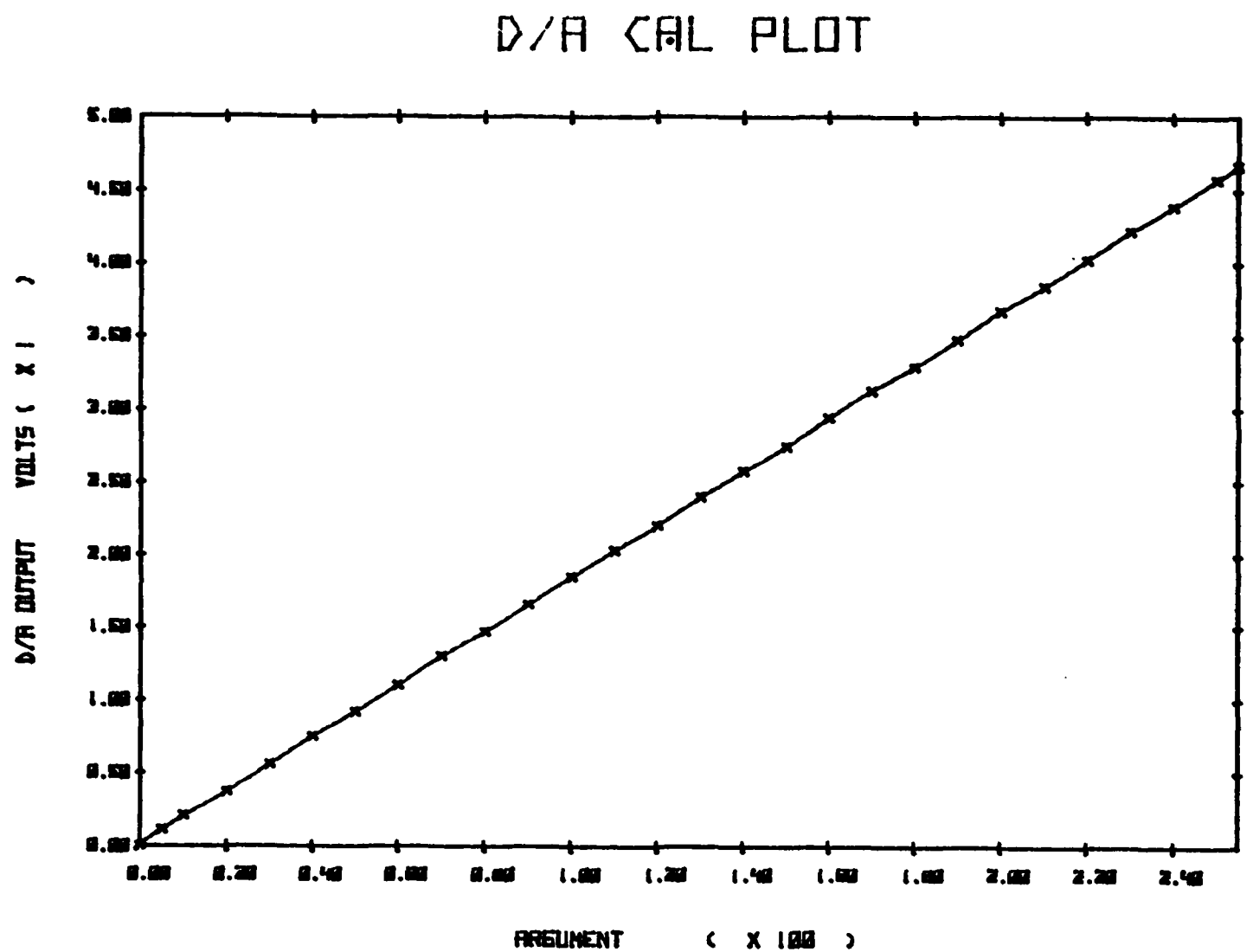
ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
138	2.758	138	139	138	138	138	140	138	138
139	2.778	139	140	139	140	139	140	139	139
140	2.795	140	141	140	140	140	141	140	140
141	2.815	141	142	141	142	141	142	141	141
142	2.836	142	143	142	142	142	143	141	142
143	2.857	143	144	143	143	143	144	143	143
144	2.858	143	144	143	144	143	144	143	143
145	2.879	144	145	145	145	144	145	144	144
146	2.899	146	146	145	146	145	146	145	145
147	2.920	147	147	146	146	146	147	146	146
148	2.937	147	148	147	148	147	148	147	147
149	2.957	149	149	148	149	148	149	148	148
150	2.978	150	150	149	150	149	150	149	149
151	2.998	150	151	150	151	150	151	150	150
152	3.018	151	152	151	152	151	152	151	151
153	3.038	153	153	152	153	152	154	152	152
154	3.059	154	154	153	154	154	154	153	153
155	3.079	155	155	155	155	154	155	154	154
156	3.096	155	156	155	155	155	156	155	155
157	3.117	156	158	156	157	156	157	156	156
158	3.137	158	158	157	158	157	158	157	157
159	3.158	159	159	158	159	158	159	158	158
160	3.190	160	161	160	160	160	161	159	160
161	3.210	162	162	161	161	161	162	161	160
162	3.230	162	163	162	162	162	163	161	162
163	3.250	163	164	163	163	163	164	163	163
164	3.268	164	164	164	164	164	165	163	163
165	3.289	165	165	165	165	165	166	164	164
166	3.309	166	167	166	166	166	167	165	166
167	3.330	167	167	167	167	167	168	166	167
168	3.350	168	169	168	168	168	169	167	167
169	3.370	169	170	169	169	169	170	168	169
170	3.390	171	171	170	170	170	171	170	170
171	3.411	171	172	171	171	171	172	170	171
172	3.428	172	172	172	172	172	173	171	171
173	3.448	173	173	173	173	173	174	172	172
174	3.468	174	174	174	174	174	175	173	173
175	3.489	175	175	175	175	175	176	174	174
176	3.491	175	175	175	175	175	176	174	174
177	3.511	176	177	176	177	176	177	176	175
178	3.532	177	178	177	177	177	178	176	177
179	3.552	178	179	178	179	178	179	177	178
180	3.569	179	179	179	180	179	180	178	179
181	3.589	180	180	180	180	180	181	179	179
182	3.610	181	181	181	182	181	182	180	181
183	3.631	182	183	182	182	182	183	181	182

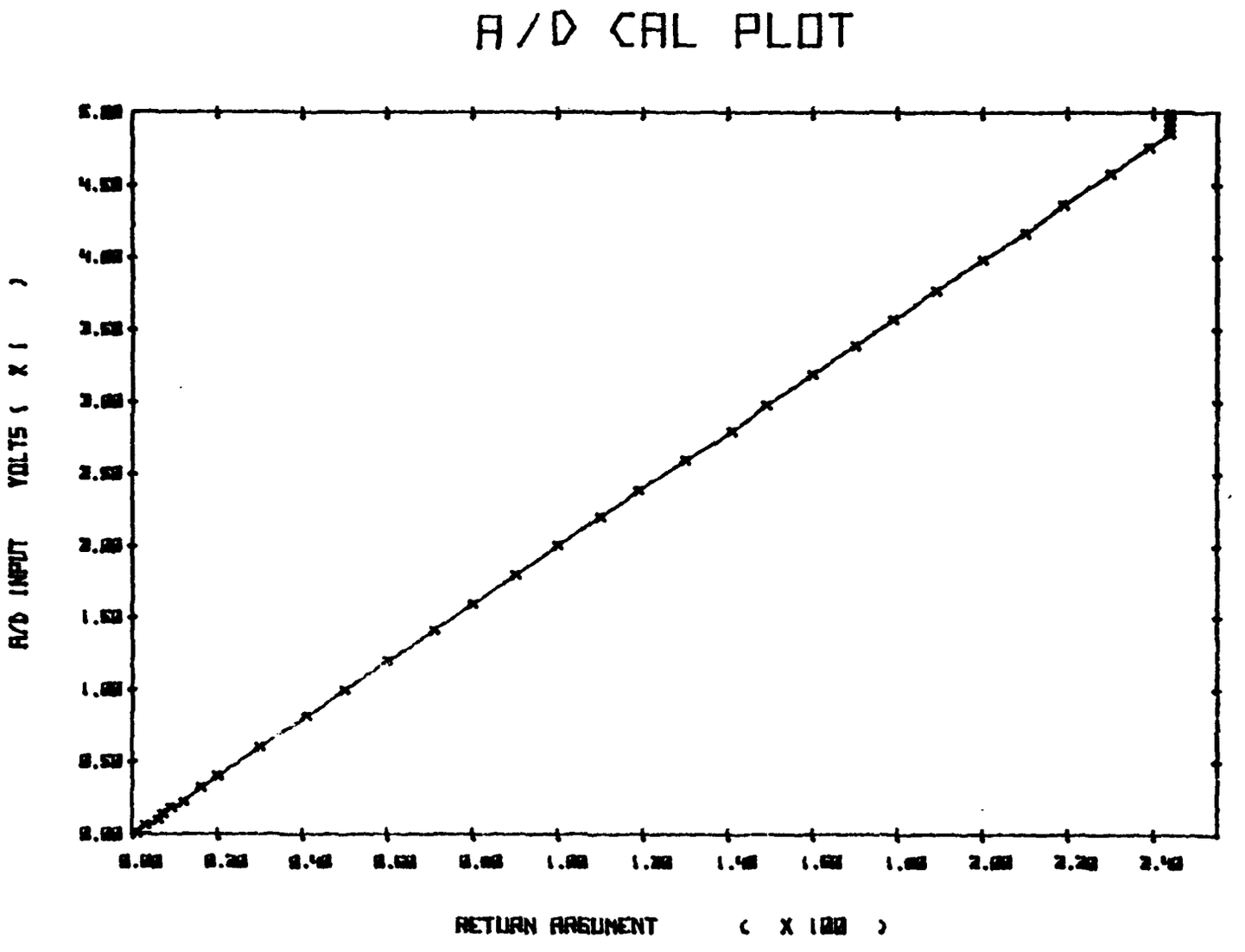
PAGE 5
UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
184	3.651	183	183	183	183	183	184	182	183
185	3.671	184	185	184	184	184	185	183	184
186	3.692	186	186	185	185	185	186	184	185
187	3.712	186	187	186	186	186	187	185	186
188	3.729	187	188	187	188	187	188	186	186
189	3.749	188	188	188	188	188	189	187	187
190	3.770	189	189	189	189	189	190	188	189
191	3.790	190	190	190	190	190	191	189	190
192	3.826	192	191	192	193	192	193	191	192
193	3.847	193	194	193	194	193	194	192	193
194	3.868	195	195	194	195	194	195	194	194
195	3.888	195	196	195	196	195	196	195	195
196	3.905	196	197	196	196	196	198	196	196
197	3.926	198	198	197	198	197	198	196	197
198	3.946	198	199	198	198	198	199	198	198
199	3.967	199	200	199	200	199	200	198	199
200	3.987	200	201	200	200	200	201	200	200
201	4.007	201	202	201	202	202	202	200	202
202	4.028	202	203	202	203	202	203	202	202
203	4.048	203	204	203	204	203	204	202	204
204	4.065	204	205	204	204	204	205	203	204
205	4.085	205	206	205	205	206	206	204	205
206	4.106	206	207	206	207	206	207	205	206
207	4.126	207	208	207	207	207	208	207	207
208	4.128	207	208	207	208	207	208	207	207
209	4.149	208	209	208	209	208	209	208	208
210	4.169	209	210	210	210	209	210	209	209
211	4.190	211	211	210	211	210	211	210	210
212	4.207	211	212	211	212	211	212	211	211
213	4.227	212	213	212	213	212	213	212	212
214	4.247	213	214	213	214	213	214	213	213
215	4.268	214	215	214	215	214	215	214	214
216	4.288	215	216	215	216	215	216	215	215
217	4.308	216	217	216	217	216	217	216	216
218	4.329	217	218	217	218	218	218	217	217
219	4.349	219	219	218	219	218	219	218	218
220	4.366	220	220	219	219	219	220	219	219
221	4.386	220	221	220	221	220	221	219	220
222	4.407	221	222	221	222	221	222	221	221
223	4.427	222	223	222	223	222	223	222	222
224	4.459	224	225	224	224	223	225	223	223
225	4.480	225	225	225	225	225	226	224	224
226	4.500	226	227	226	226	226	227	225	225
227	4.521	227	228	227	227	227	228	226	227
228	4.538	228	229	228	228	227	229	227	227
229	4.558	229	229	229	229	229	230	228	228

PAGE 6
UID A/D & D/A CALIBRATION TABLE

ARG	VOLTS	CH 0	CH 1	CH 2	CH 3	CH 4	CH 5	CH 6	CH 7
230	4.579	230	230	230	230	230	231	229	229
231	4.599	231	232	231	231	230	232	230	231
232	4.619	232	233	232	232	232	234	231	231
233	4.640	233	234	233	233	232	234	232	233
234	4.660	234	235	234	234	234	235	233	233
235	4.680	235	235	235	235	235	236	234	235
236	4.697	236	236	236	236	235	237	236	235
237	4.718	237	237	237	237	236	238	236	236
238	4.738	238	238	238	238	238	239	237	237
239	4.759	239	239	239	239	238	240	238	238
240	4.761	239	240	239	239	239	240	238	239
241	4.781	240	241	240	240	240	241	239	239
242	4.802	241	242	241	242	241	242	240	241
243	4.922	242	243	242	243	242	243	241	242
244	4.839	243	244	243	243	243	244	243	243
245	4.860	244	245	244	244	244	245	243	244
246	4.880	245	246	244	244	244	244	245	245
247	4.900	246	246	244	244	244	244	245	246
248	4.921	246	246	244	244	244	244	247	246
249	4.941	246	246	244	244	244	244	246	246
250	4.962	246	246	244	244	244	244	246	246
251	4.982	246	246	244	244	244	244	246	246
252	4.999	246	246	244	244	244	244	246	246
253	5.019	246	246	244	244	244	244	247	246
254	5.040	246	246	244	244	244	244	247	246
255	5.060	246	246	244	244	244	244	247	246


D/A CAL PLOT
D/A OUTPUT VOLTS (X 1)
ARGUMENT (X 100)


A/D CAL PLOT
A/D INPUT VOLTS (X 1)
RETURN ARGUMENT (X 100)

APPENDIX H

SYSTEM VALIDATION RESULTS

The TRS-80/UID was validated by evaluating a NACA 66(215)-216 wing section in the 32" x 45" academic wind tunnel as described in Section V. The results are compared against manually collected data from the tunnel manometer array.

Airfoil Characteristics

NACA 66(215)-216

chord = 12"

leading edge = port 1 (radius = 1.58%C)

trailing edge = port 36 (thickness = .3%C)

upper surface			lower surface		
port	station (% C)	ordinate (% C)	port	station (% C)	ordinate (% C)
2	0.33	1.20	3	0.33	0.90
4	0.83	1.68	5	0.83	1.28
6	1.25	1.97	7	1.00	1.43
8	2.25	2.62	9	2.18	1.93
10	3.63	3.27	11	3.17	2.32
12	5.35	3.92	13	4.83	2.77
14	10.0	5.38	15	9.80	3.68
16	16.0	6.78	17	15.3	4.68
18	20.3	7.55	19	20.0	5.27
20	30.7	8.80	21	30.4	6.10
22	40.4	9.38	23	40.2	6.48
24	50.1	9.44	25	50.0	6.50
26	55.0	9.24	27	55.0	6.32
28	60.0	8.82	29	60.0	6.04
30	70.0	7.17	31	70.0	4.84
32	80.0	4.64	33	80.2	3.04
34	90.0	1.96	35	90.2	1.20

Wind Tunnel Description

Type: AEROLAB 90 Series, 32" x 45", lowspeed; located in Halligan Hall, Naval Postgraduate School, Monterey, California 93940

Pressure Transducer: capacitive, amplified to +/-15 volts analog output

Scanivalve: 48 channel (1-48) single step & home (ch 48)

Parameters:

P0 = ambient static pressure

P1 = tunnel plenum static pressure

P2 = test section static pressure

V = test section velocity

RHO= density

Pressure/Velocity Relationship:

V = SQR((2*(P2-P1))/RHO)

Manometer Array Configuration

40 tubes

S.G. of Fluid = .939

Tube	Parameter	Scanivalve Channel	Test Section Port
1	PO	NC	NC
2	PO	NC	NC
3	test port	4	1
4	test port	5	2
5	test port	6	3
6	test port	7	4
7	test port	8	5
8	test port	9	6
9	test port	10	7
10	test port	11	8
11	test port	12	9
12	test port	13	10
13	test port	14	11
14	test port	15	12
15	test port	16	13
16	test port	17	14
17	test port	18	15
18	test port	19	16
19	test port	20	17
20	test port	21	18
21	test port	22	19
22	test port	23	20
23	test port	24	21
24	test port	25	22
25	test port	26	23
26	test port	27	24
27	test port	28	25
28	test port	29	26
29	test port	30	27
30	test port	31	28
31	test port	32	29
32	test port	33	30
33	test port	34	31
34	test port	35	32
35	test port	36	33
36	test port	37	34
37	test port	38	35
38	test port	39	36
39	PO	NC	NC
40	PO	NC	NC

Scanivalve Configuration

48 pressure channel input/single channel output

Electrical control inputs (make & break):

STEP: advance to next channel

HOME: advance to channel 48

Electrical control outputs (TTL):

7 bit BCD encoded channel address (inverted)

Scanivalve Configuration
(cont)

Channel	Parameter	Test Section Port	Manometer Tube
1	P0	NC	NC
2	P1	NC	NC
3	P2	NC	NC
4	test port	1	2
5	test port	2	3
6	test port	3	4
7	test port	4	5
8	test port	5	6
9	test port	6	7
10	test port	7	8
11	test port	8	9
12	test port	9	10
13	test port	10	11
14	test port	11	12
15	test port	12	13
16	test port	13	14
17	test port	14	15
18	test port	15	16
19	test port	16	17
20	test port	17	18
21	test port	18	19
22	test port	19	20
23	test port	20	21
24	test port	21	22
25	test port	22	23
26	test port	23	24
27	test port	24	25
28	test port	25	26
29	test port	26	27
30	test port	27	28
31	test port	28	29
32	test port	29	30
33	test port	30	31
34	test port	31	32
35	test port	32	33
36	test port	33	34
37	test port	34	35
38	test port	35	36
39	test port	36	37
40	P1	NC	NC
41	P2	NC	NC
42	P0	NC	NC
42-48	NC	NC	NC

Manometer Array Observation

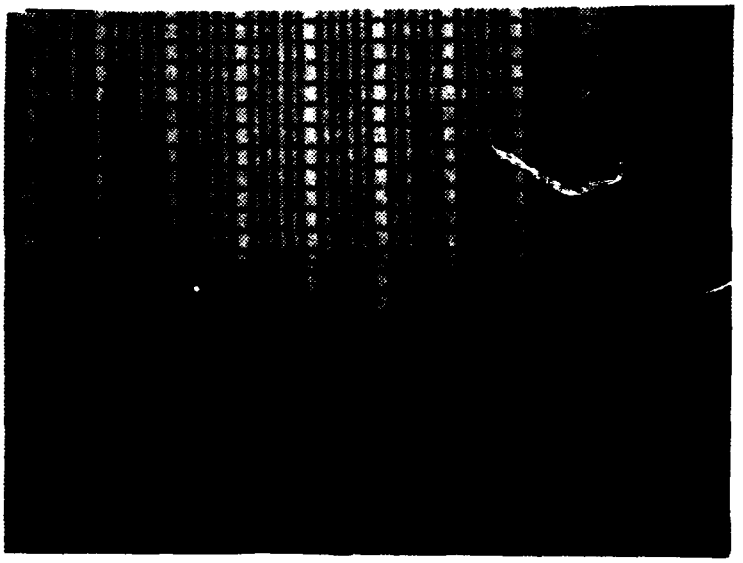
Test Run: 22 OCT 1982

Nominal Tunnel Velocity = 90 mph

Ambient Temperature = 62 DEG F

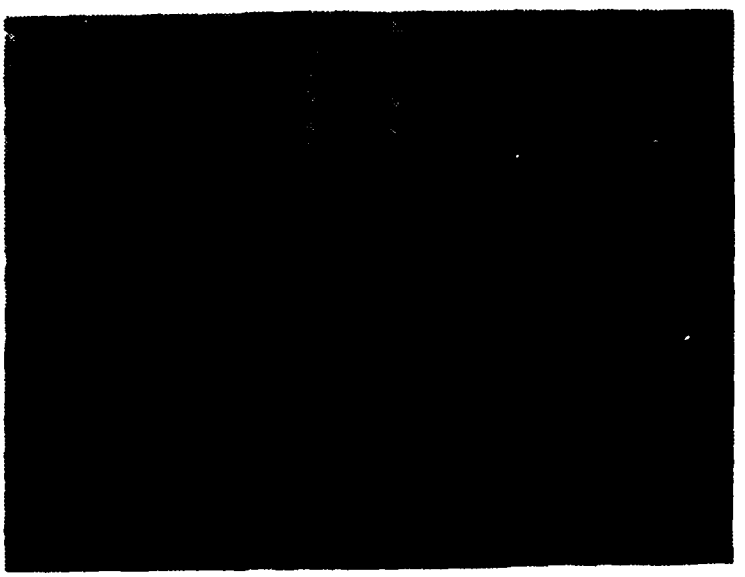
Test Section: NACA 66(215)-216

Angle of Attack = 10 DEG



Manometer Array
NACA 66(215)-216
90 mph
alpha = 10 DEG

Manometer Array Observation
(cont)



Manometer Array
NACA 66(215)-216
zero velocity
alpha = 10 DEG

Manometer Array Observation
(cont)

Tabulation of Results
NACA 66(215)-216
90 mph / alpha = 10 DEG

Manometer Tube	Parameter	Scanivalve Channel	Test Port	Reading (in H20)
1	P0	NC	NC	31.6
2	P0	NC	NC	31.6
3	test port	4	1	37.8
4	test port	5	2	45.3
5	test port	6	3	31.8
6	test port	7	4	45.8
7	test port	8	5	28.5
8	test port	9	6	43.9
9	test port	10	7	27.7
10	test port	11	8	42.8
11	test port	12	9	27.6
12	test port	13	10	41.8
13	test port	14	11	27.9
14	test port	15	12	38.3
15	test port	16	13	28.4
16	test port	17	14	36.9
17	test port	18	15	29.4
18	test port	19	16	36.6
19	test port	20	17	30.0
20	test port	21	18	36.2
21	test port	22	19	30.4
22	test port	23	20	35.5
23	test port	24	21	31.0
24	test port	25	22	35.0
25	test port	26	23	31.5
26	test port	27	24	34.6
27	test port	28	25	31.8
28	test port	29	26	33.2
29	test port	30	27	32.0
30	test port	31	28	32.8
31	test port	32	29	32.0
32	test port	33	30	32.6
33	test port	34	31	31.9
34	test port	35	32	32.0
35	test port	36	33	31.6
36	test port	37	34	31.9
37	test port	38	35	30.8
38	test port	39	36	31.9
39	P0	NC	NC	31.6
40	P0	NC	NC	31.6

TRS-80/UID Results

ALL READINGS IN INCHES H20

P0 = 31.552
P1 = 27.6435
P2 = 31.3581

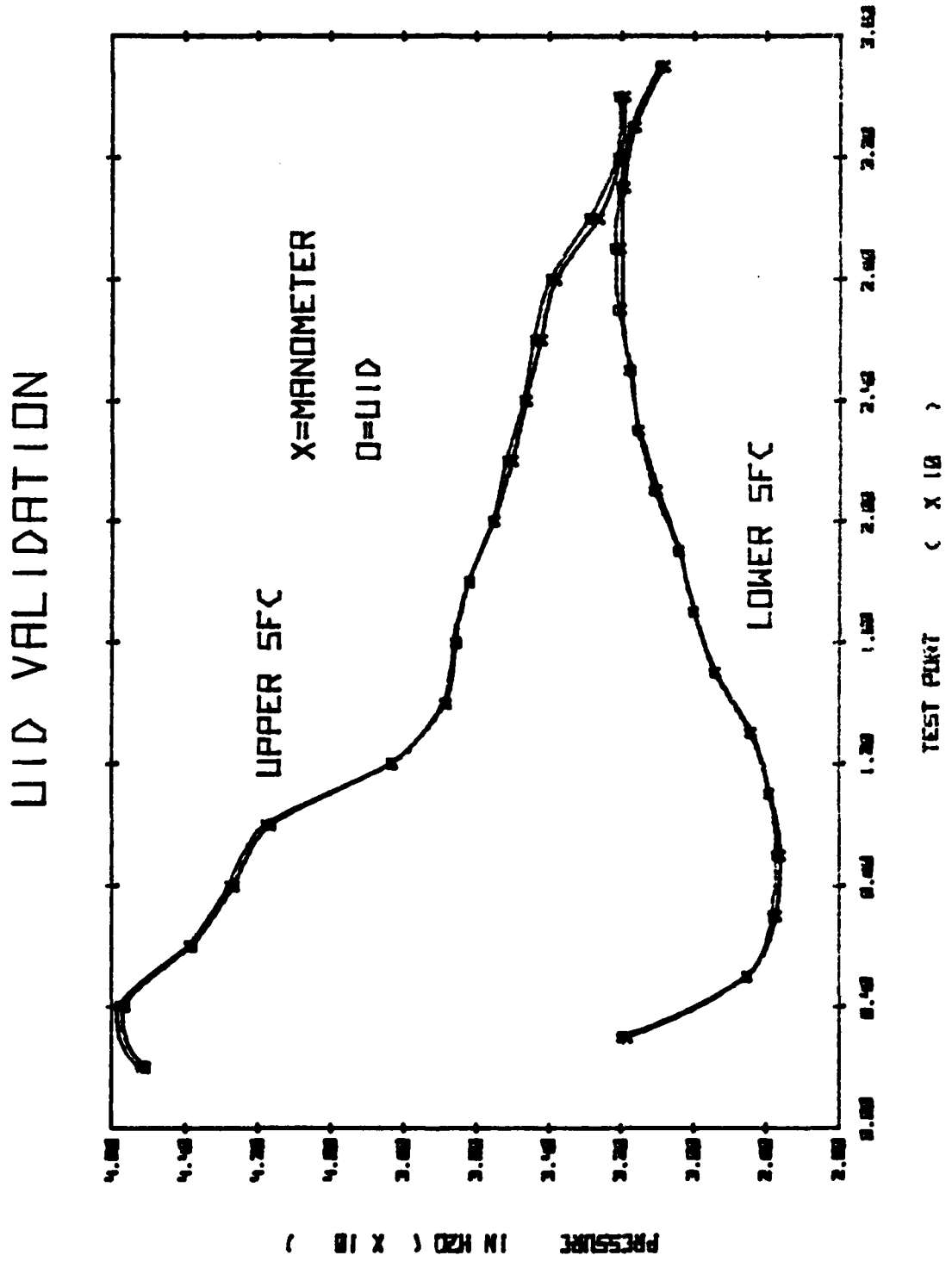
PORT	VALUE
1	3.775E+01
2	4.509E+01
3	3.199E+01
4	4.566E+01
5	2.854E+01
6	4.381E+01
7	2.782E+01
8	4.266E+01
9	2.774E+01
10	4.615E+01
11	2.793E+01
12	3.834E+01
13	2.848E+01
14	3.683E+01
15	2.945E+01
16	3.655E+01
17	3.002E+01
18	3.620E+01
19	3.043E+01
20	3.555E+01
21	3.111E+01
22	3.517E+01
23	3.156E+01
24	3.469E+01
25	3.174E+01
26	3.440E+01
27	3.210E+01
28	3.397E+01
29	3.217E+01
30	3.287E+01
31	3.201E+01
32	3.211E+01
33	3.170E+01
34	3.209E+01
35	3.094E+01
36	3.195E+01

NACA 66(215)-216
90 mph / alpha = 10 DEG

Cross Tabulation

UID vs Manometer
NACA 66(215)-216
90 mph / alpha = 10 DEG

Parameter	Manometer Tube	Scanivalve Channel	Reading (In. H20)	Reading (In. H20)
P0	1	1	31.6	31.55
P1	NC	2	NA	27.64
P2	NC	3	NA	31.36
port 1	3	4	37.8	37.75
port 2	4	5	45.3	45.09
port 3	5	6	31.8	31.99
port 4	6	7	45.8	45.66
port 5	7	8	28.5	28.54
port 6	8	9	43.9	43.81
port 7	9	10	27.7	27.82
port 8	10	11	42.8	42.66
port 9	11	12	27.6	27.74
port 10	12	13	41.8	41.65
port 11	13	14	27.9	27.93
port 12	14	15	38.3	38.34
port 13	15	16	28.4	28.48
port 14	16	17	36.9	36.83
port 15	17	18	29.4	29.45
port 16	18	19	36.6	36.55
port 17	19	20	30.0	30.02
port 18	20	21	36.2	36.20
port 19	21	22	30.4	30.43
port 20	22	23	35.5	35.55
port 21	23	24	31.0	31.11
port 22	24	25	35.0	35.17
port 23	25	26	31.5	31.56
port 24	26	27	34.6	34.69
port 25	27	28	31.8	31.74
port 26	28	29	34.2	34.40
port 27	29	30	32.0	32.10
port 28	30	31	33.8	33.97
port 29	31	32	32.0	32.17
port 30	32	33	32.6	32.87
port 31	33	34	31.9	32.01
port 32	34	35	32.0	32.11
port 33	35	36	31.6	31.70
port 34	36	37	31.9	32.09
port 35	37	38	30.8	30.94
port 36	38	39	31.9	31.95


UID VALIDATION
X=MANOMETER
O=UID
UPPER SFC
LOWER SFC
TEST POINT (X 10)
PRESSURE IN H2O (X 10)

```
  1 DATA"*************************************************************"
  2 DATA"***                                                       ***"
  3 DATA"***              UID VALIDATION PROGRAM                   ***"
  4 DATA"***                                                       ***"
  5 DATA"***              BY LCDR W. B. BROWN, JR                  ***"
  6 DATA"***                  20OCT82/REV 1.0                      ***
  7 DATA"***                                                       ***"
  8 DATA"*************************************************************"
  9 DATA" "
 10 CLS:FOR T=1 TO 9:READ T$:PRINT T$:NEXT:PRINT"PRESS ENTER TO CONTINUE
    "
 11 GOSUB 32100:'CR
 12 CLS
 13 PRINT"RUNNING"
 20 CLEAR 5000
 22 DIM M$(15,2),SP(40),NP(48),PP(3,'
100 'ALG PROGRAM DRIVER
102       GOSUB 10000,'INITIALIZE
103       GOSUB 7000:'MENU
104       GOSUB 200:'MENU DRIVER
106       GOSUB 54000:'DUMP TO DISK
110       CLS
112       PRINT"END OF PROGRAM"
114       PRINT
116       END
199 'END SLG

1000 'SBR SCANIVALVE TO PORT DRIVER
1010       PRINT
1020       PRINT"SCANIVALVE TO MATCH PORT";
1030       INPUT SP
1032       'IF SP>0 AND SP<=NP THEN
1034            IF NOT((SP>0)AND(SP<=NP)) THEN 1030
1040            GOSUB 31400:'ADVANCE SCANIVALVE TO PORT
1050            PRINT
```

```
1060            GOSUB 3000:'REPORT ADDRESS
1062            GOTO 1090
1064        'ELSE
1070            PRINT"INVALID PORT ADDRESS"
1090        'END IF
1098        RETURN
1099 'END SBR

2000 'SBR SCANIVALVE TO ADDR
2010        PRINT
2020        PRINT"SCANIVALVE TO WHICH ADDRESS"
2030        INPUT SA
2040        GOSUB 31300:'ADVANCE SCANIVALVE TO ADDRESS
2050        PRINT
2060        GOSUB 3000:'REPORT ADDRESS
2998        RETURN
2999 'END SBR

3000 'SBR REPORT ADDRESS
3010        GOSUB 31200:'READ SCANIVALVE ADDRESS
3020        T=0
3030        'DO UNTIL SV=SP(T) OR T>40
3040            T=T+1
3050            IF(NOT((SV=SP(T))OR(T>40)))THEN 3030
3060        'END DO
3065        IF T>40 THEN T=0
3070        PRINT"SCANIVALVE AT ADDRESS";SV;" PORT";T
3998        RETURN
3999 'END SBR

4000 'SBR CALIBRATION DRIVER
4002        CLS
4004        PRINT"CALIBRATING WIND TUNNEL TO 110"
4010        PRINT
4020        PRINT"ADJUST WIND TUNNEL TO DESIRED SPEED"
4030        PRINT"PRESS ENTER TO CONTINUE"
```

```
4040        GOSUB 32100:'CR
4050        PRINT
4060        INPUT"ENTER PORT OF HIGHEST READING";SU
4070        PRINT
4080        INPUT"ENTER PORT OF LOWEST READING";SL
4090        PRINT
4100        GOSUB 31500:'CALIBRATE SCANIVALVE TRANSDUCER
4110        GOSUB 31900:'SCALE SCANIVALVE TRANSDUCER
4998        RETURN
4999 'END SBR

5000 'SBR TEST RUN DRIVER
5010        CLS
5020        PRINT"COMMENCING TEST RUN"
5030        PRINT
5040        INPUT"ENTER DELAY FACTOR";SD
5050        PRINT
5060        V$="IS TUNNEL CALIBRATED"
5070        GOSUB 32200:'Y/N
5080        IF KB$="N" GOSUB 4000:'CALIBRATION DRIVER
5090        PRINT
5100        GOSUB 52000:'READ P0,P1,P2
5110        GOSUB 51000:'READ TEST PORTS
5120        PRINT
5130        PRINT"RUN COMPLETE"
5140        PRINT"OK TO POWER DOWN TUNNEL"
5150        PRINT
5998        RETURN
5999 'END SBR

6000 'SBR OUTPUT DRIVER
6010        V$="WANT OUTPUT"
6020        GOSUB 32200:'Y/N
6030        IF KB$="Y" GOSUB 53000:'OUTPUT SUB DRIVER
6998        RETURN
6999 'END SBR
```

```
7000 'SBR MENU
7010       PRINT
7020       PRINT"VALID COMMANDS"
7030       T=0:T1$="END"
7040       'DO WHILE M$(T,1)<>T1$
7045            IF(NOT(M$(T,1)<>T1$))THEN 7080
7050            PRINT TAB(5);M$(T,1)
7060            T=T+1
7070            GOTO 7040
7080       'END DO
7998       RETURN
7999 'END SBR

8000 'SBR HOME SCANIVALVE DRIVER
8010       GOSUB 31600:'HOME SCANIVALVE
8998       RETURN
8999 'END SBR

9000 'SBR ADVANCE SCANIVALVE DRIVER
9010       GOSUB 31700:'ADVANCE SCANIVALVE
9020       GOSUB 3000:'ADDRESS
9998       RETURN
9999 'END SBR

10000 'SBR INITIALIZE
10010       GOSUB 50000:'INITIALIZE MENU
10020       GOSUB 50100:'INITIALIZE SCANIVALVE INFO
10030       GOSUB 50200:'INITIALIZE MISC
10998       RETURN
10999 'END SBR

11000 'SBR READ A/D
11010       CLS
11020       PRINT
11030       PRINT
11040       PRINT"READING A/D CHANNEL";SC
11050       PRINT"PRESS ENTER WHEN READY"
```

```
11060       PRINT"TO RETURN TO MENU"
11070       PRINT@0," ",SU$(SC)
11080       'DO UNTIL KB$=CHR$(13)
11090            GOSUB 31100:'READ TRANSDUCER
11092            VI=P
11100            GOSUB 31800:'SCALE
11110            KB$=INKEY$
11120            PRINT @0,"          ";:PRINT@0,VO
11130            IF(NOT(KB$=CHR$(13)))THEN 11080
11140       'END DO
11142       CLS
11998       RETURN
11999 'END SBR

50000 'SBR INITIALIZE MENU
50002       RESTORE
50004       T1$="MENU LIBRARY"
50006       'DO UNTIL  T$=T1$
50008            READ T$
50010            IF(NOT(T$=T1$))THEN 50006
50012       'END DO
50014       T=1
50016       READ M$(T,1)
50018       T1$="END"
50020       'DO WHILE M$(T,1)<>T1$
50022            IF(NOT(M$(T,1)<>T1$))THEN 50032
50024            READ M$(T,2)
50026            T=T+1
50028            READ M$(T,1)
50030            GOTO 50020
50032       'END DO
50098       RETURN
50099 'END SBR
50100 'INITIALIZE SCANIVALVE INFO
50110 SC=0
```

```
50120 NP=36
50130 'DO FOR T=1 TO NP
50132       FOR T=1 TO NP
50140       SP(T)=3+T
50150       NEXT
50160 'END DO
50170 PP(0,2)=1
50172 SH=0:SS=1
50174 SN=10
50180 PP(1,2)=2
50190 PP(2,2)=3
50198 RETURN
50199 'END SBR
50200 'SBR INITIALIZE MISC
50201       LM=15
50202       PL=50
50204       MG$=CHR$(140)+STRING$(5,13)
50206       LM=15
50298       RETURN
50299 'END SBR
50300 'SBR MENU LIBRARY
50302       DATA"MENU LIBRARY"
50304       DATA"SV TO PORT","1"
50306       DATA"SV TO ADDR","2"
50307       DATA"HOME SV","8"
50308       DATA"ADV SV","9"
50309       DATA"ADDRESS","3"
50310       DATA"CALIBRATE","4"
50312       DATA"TEST RUN","5"
50313       DATA"READ A/D","11"
50314       DATA"OUTPUT","55"
50316       DATA"MENU","7"
50397       DATA"END"
50398       RETURN
50399 'END SBR
```

```
52000 'SBR READ P0,P1,P2
52010       'DO FOR I=0 TO 2
52020            FOR I=0 TO 2
52030            SA=PP(I,2)
52040            GOSUB 31300:'ADVANCE SCANIVALVE TO ADDRESS
52050            FOR T=1 TO 270*SD:NEXT:'DELAY
52060            GOSUB 31100:'READ SCANIVALVE TRANSDUCER
52072            VI=P
52074            GOSUB 31800:'SCALE
52075            PRINT
52076            PP(I,1)=V0
52077            GOSUB 3000:'ADDRESS
52078            PRINT"P";RIGHT$(STR$(I),1);" =";PP(I,1);SU$(SC)
52080            NEXT
52090       'END DO
52998       RETURN
52999 'END SBR
53000 'OUTPUT SUBDRIVER
53010 LPRINT CHR$(140);STRING$(5,13)
53012 LPRINT TAB(LM);"ALL READINGS IN ";SU$(SC)
53014 LPRINT
53020 LPRINT TAB(LM);"P0 =";PP(0,1)
53030 LPRINT TAB(LM);"P1 =";PP(1,1)
53040 LPRINT TAB(LM);"P2 =";PP(2,1)
53050 LPRINT
53070 LPRINT TAB(LM);"PORT","      ";"VALUE"
53072 LPRINT
53080 'DO FOR I=1 TO NP
53090       FOR I=1 TO NP
53092       LPRINT TAB(LM);
53100       LPRINT USING "##                    ##.####^^^^";I,P(I)
53110       NEXT
53120 'END DO
53130 LPRINT CHR$(140);STRING$(5,13)
53998 RETURN
53999 'END SBR
```

```
51000 'SBR READ TEST PORTS
51010       'DO FOR I=1 TO NP
51020             FOR I=1 TO NP
51030             SP=I
51040             GOSUB 31400:'ADVANCE SCANIVALVE TO PORT
51050             FOR T=1 TO 270*SD:NEXT:'DELAY
51060             GOSUB 31100:'READ SCANIVALVE TRANSDUCER
51061             VI=P:GOSUB 31800:P(I)=VO:PRINT:T$=STR$(I):T=LEN(T$):T$=RIG
                     HT$(T$,T-1):PRINT"P(";T$;") =";P(I);SU$(SC)
51070             GOSUB 3000:'REPORT ADDRESS
51090             NEXT
51100       'END DO
51998       RETURN
51999 'END SBR

54000 'SBR DUMP TO DISK
54010       CLS
54020       V$="WANT TO SAVE DATA TO DISK"
54030       GOSUB 32200:'Y/N
54040       'IF YN$="Y" THEN
54050             IF(NOT(YN$="Y")) THEN XXXXX
54060             PRINT
54070             INPUT"NAME THE FILE (8 CHAR MAX)";T$
54080             IF LEN(T$)>8 THEN T$=LEFT$(T$,8)
54090             T$=T$+"/TXT"
54100             OPEN"O",1,T$
54110             PRINT#1,SU$(SC)
54112             PRINT SU$(SC)
54120             'DO FOR T=0 TO 2
54130                   FOR T=0 TO 2
54140                   T1$=STR$(PP(I,T))
54150                   PRINT#1,T1$
54152                   PRINT T1$
54160                   NEXT
```

```
54170             'END DO
54180             T1$=STR$(NP)
54190             PRINT#1,T1$
54192             PRINT T1$
54200             'DO FOR T=1 TO NP
54210                  FOR T=1 TO NP
54220                  T1$=STR$(P(T))
54230                  PRINT#1,T1$
54232                  PRINT T1$
54240                  NEXT
54250             'END DO
54260             PRINT:PRINT
54270             PRINT"FILE NOW ON DISK"
54280        'END IF
54998        RETURN
54999 'END SBR

60000 OUT 1,2
60010 FOR T=1 TO 50:NEXT
60020 OUT 1,0
60030 RETURN
61000 OUT 1,1
61010 FOR T=1 TO 50:NEXT
61020 OUT 1,0
61030 RETURN
62000 M(0)=.02:SU$(0)="VOLTS":RETURN
```

APPENDIX I

GLOSSARY

1. Address bus: that group of lines from the CPU which it uses to vector data to/from memory locations by means of binary electrical signals.

2. Assembly language (code): a low level computer language which uses mnemonics and higher order number systems to represent the binary operation codes and absolute addresses of the CPU but is otherwise a one for one translation of the CPU machine code.

3. A/D: analog to digital

4. Baud: the data transmission rate expressed in bits per second.

5. BCD: Binary Encoded Decimal. A 4 bit binary bit encoding representing the decimal digits 0 to 9.

6. BIT: Binary digit: single unit of information representing a one/zero, on/off, or yes/no state.

7. Bug: any software or hardware error/malfunction that causes the computer/program to operate in a manner other than intended.

8. Byte: a group of eight bits which are processed as a single quantity.

9. CPU: Central Processing Unit. The area of a computer which computes all logic and arithmetic functions and sequences the flow of the program.

10. CRT: Cathode Ray Tube. Used as the generic name for television type I/O devices.

11. Data bus: that group of lines from the CPU on which the information that is processed is vectored from CPU, memory, and I/O devices as binary electrical signals.

12. DIP: Dual In-line Package. The standard integrated circuit housing characterized by two rows of symmetric pins in a low profile, rectangular package.

13. D/A: digital to analog; the inverse of A/D.

14. D/D: Digital to Digital, from one logic family to another or from the clocked data bus under CPU control to a latched I/O device.

15. HEX (hexadecimal): the base 16 number system. The digits 10 to 15 are represented by the letters A to F. Thus, 12 decimal = C hex; FF hex = 255 decimal.

16. I/O: input/output.

17. K: a suffix which indicates a group of 1024 (2**10) items as in '16K of memory' meaning 16384 memory locations.

18. LED: Light Emitting Diode. A type of diode that emits visible light in its forward biased condition; used frequently as an indicator because of low voltage/ power requirements.

19. Machine language (code): the binary bit patterns, represented by electrical signals, used by the CPU to carry out its programmed functions.

20. Nibble: the upper or lower four bits of a byte.

21. OPAMP: Operational Amplifier. An extremely stable linear amplifier capable of precise gain.

22. RAM: Random Access Memory. Volatile memory the contents of which can be changed by the CPU.

23. ROM: Read Only Memory. Non-volatile memory the contents of which may not be changed by the CPU but may be Read Only.

24. Software: the program that resides in the microprocessor memory.

25. Structured Programming: a modular programming technique characterized by top to bottom program flow with transfer out of the statement order not allowed except within defined, logical constructs.

26. TTL: Transistor Transistor Logic. A family of digital logic and IC's characterized by a +5 volt monopolar power supply, high noise immunity, high speed (20 MHz typical), and with directly cascadeable I/O. A signal above 2.4 volts represents an on or '1' state; below .8 volts, an off or '0' state.

27. Word: a group of one or more bytes which the CPU is capable of treating as a single quantity; the binary order of the data bus expressed in bytes or bits.

28. Z80: a third generation microprocessor using an eight bit word, sixteen bit address bus (64K), and requiring a single +5 volt supply voltage.

LIST OF REFERENCES

1. Hogan, T., CP/M User's Guide, pp. 1-3, Osborne/McGraw Hill, 1982.

2. Kepner, Terry, "Feedback Loop," 80 Microcomputing, p. 82, 1001001 Inc., June/July 1982.

3. Busch, David D., "Broadening the TRS-80 Horizon," 80 Microcomputing, p. 298, 1001001 Inc., March 1982.

4. Commander, Jake, "Commander 80," 80 Microcomputing, p. 74, 1001001 Inc., March 1982.

5. Titus, Johnathan A., TRS-80 Interfacing Book 1, pp. 47-60, Howard W. Sams, Inc., 1979.

6. Farvour, James, TRS-80 BASIC Decoded and Other Mysteries for the TRS-80, pp. 42-51, IJG Computer Services, 1981.

7. Lancaster, Don, TTL Cookbook, pp. 7-37, Howard W. Sams, Inc., 1974.

8. Kurtz, Thomas E., "On the Way to Standard Basic," Byte, p. 182, McGraw Hill, June 1982.

BIBLIOGRAPHY

Barden, W., TRS-80 Assembly Language Subroutines, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1982.

Barden, W., Programming Techniques for Level II BASIC, Radio Shack, Tandy Corporation, Fort Worth, Texas, 1980.

Blattner, T. and Mumford, B., Inside Level II, Mumford Microsystems, Summerland, California, 1980.

Ciarcia, S., Build Your Own Z80 Microcomputer, Byte/McGraw Hill, Peterborough, New Hampshire, 1981.

Farvour, J., Microsoft BASIC Decoded & Other Mysteries for the TRS-80, TJG Computer Services, Upland, California, 1981.

Graham, N., Introduction to Computer Science, a Structured Approach, West Publishing Co., St. Paul, Minnesota, 1979.

Green, W., 80 Micro (formerly 80 Microcomputing), 1001001, Inc., Peterborough, New Hampshire, monthly Vol. I and II.

Howe, H. S., TRS-80 Assembly Language, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1981.

Lancaster, D., TTL Cookbook, Howard W. Sams & Co., Inc., Indianapolis, Indiana, 1974.

Leventhal, L. A., Introduction to Microprocessors; Software Hardware, Programming, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1978.

Malvino, A. P., Digital Computer Electronics, McGraw Hill, New York, New York, 1977.

Peterson, G. R., Basic Analog Computation, MacMillian Company, Toronto, Ontario, 1967.

Radio Shack, Level II Reference Manual, Tandy Corporation, Fort Worth, Texas, 1979.

Radio Shack, TRSDOS & Disk BASIC Reference Manual, Tandy Corporation, Fort Worth, Texas, 1979.

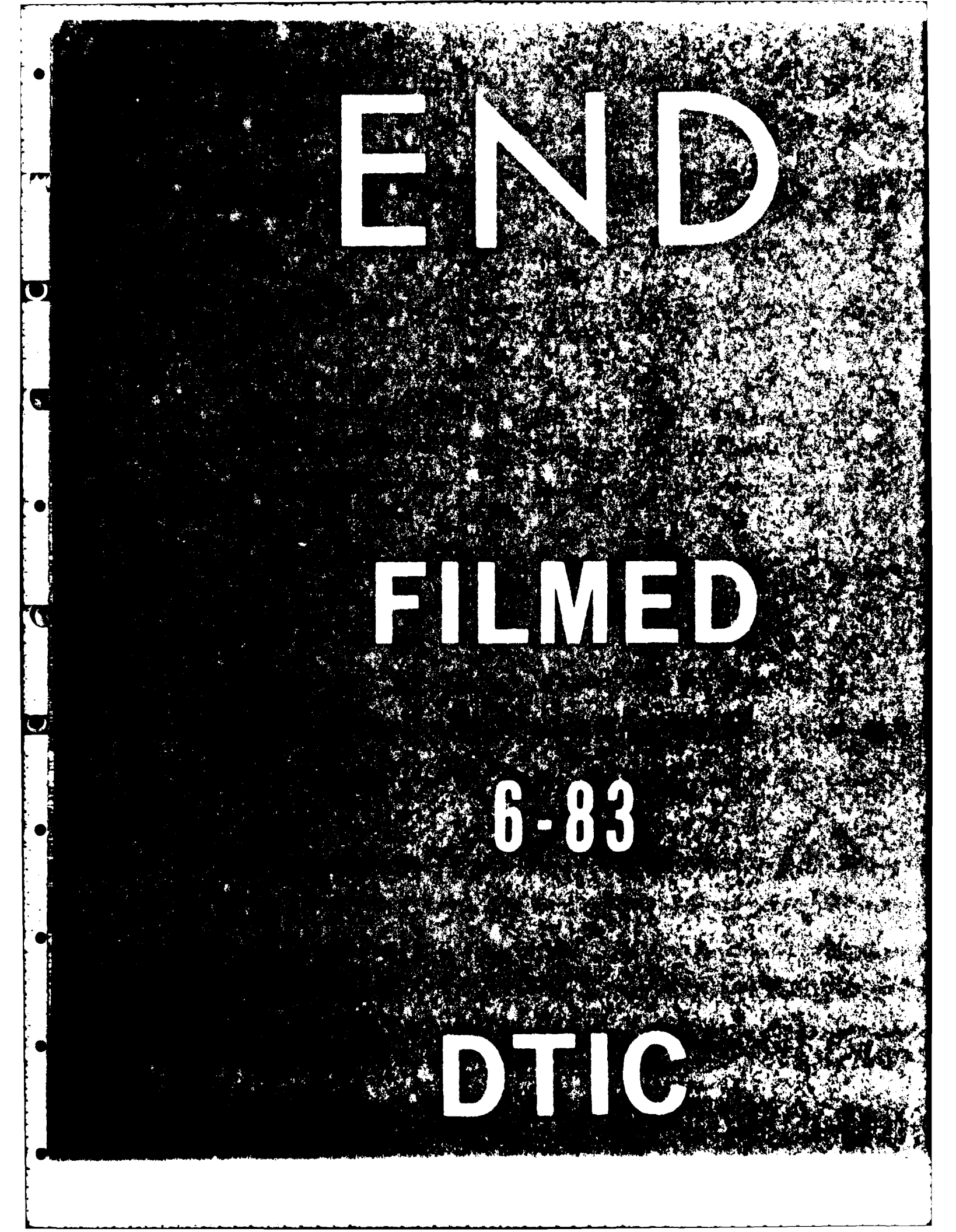
Radio Shack, TRS-80 Microcomputer Technical Reference Handbook, Tandy Corporation, Fort Worth, Texas, 1978.

Ruckdeschel, F. R., BASIC Scientific Subroutines Vol. I & II, Byte/McGraw Hill, Peterborough, New Hampshire, 1981.

Titus, J. A., TRS-80 Interfacing Books 1 & 2, Howard W. Sams & Co., Inc., Indianapolis, Indiana, 1979.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 67 Department of Aeronautics Naval Postgraduate School Monterey, California 93940	1
4.	Professor Donald M. Layton, Code 67-LN Department of Aeronautics Naval Postgraduate School Monterey, California 93940	5
5.	LCDR W. Burcher Brown, Jr., USN 1718 Gum Tree Drive Orange Park, Florida 32073	10


END
FILMED
6-83
DTIC